IDA PAPER P-2152

META-DESIGN — AN APPROACH TO THE DEVELOPMENT OF DESIGN METHODOLOGIES

J. E. Rogan, *Georgia Institute of Technology*
School of Aerospace Engineering

William E. Cralley

January 1990

Prepared for
Office of the Under Secretary of Defense for Acquisition
(Research and Advanced Technology)

Supported by
Air Force Human Resources Laboratory
Wright-Patterson AFB, Ohio

IDA

INSTITUTE FOR DEFENSE ANALYSES
1801 N. Beauregard Street, Alexandria, Virginia 22311-1772

DEFINITIONS

IDA publishes the following documents to report the results of its work.

Reports

Reports are the most authoritative and most carefully considered products IDA publishes. They normally embody results of major projects which (a) have a direct bearing on decisions affecting major programs, (b) address issues of significant concern to the Executive Branch, the Congress and/or the public, or (c) address issues that have significant economic implications. IDA Reports are reviewed by outside panels of experts to ensure their high quality and relevance to the problems studied, and they are released by the President of IDA.

Group Reports

Group Reports record the findings and results of IDA established working groups and panels composed of senior individuals addressing major issues which otherwise would be the subject of an IDA Report. IDA Group Reports are reviewed by the senior individuals responsible for the project and others as selected by IDA to ensure their high quality and relevance to the problems studied, and are released by the President of IDA.

Papers

Papers, also authoritative and carefully considered products of IDA, address studies that are narrower in scope than those covered in Reports. IDA Papers are reviewed to ensure that they meet the high standards expected of refereed papers in professional journals or formal Agency reports.

Documents

IDA Documents are used for the convenience of the sponsors or the analysts (a) to record substantive work done in quick reaction studies, (b) to record the proceedings of conferences and meetings, (c) to make available preliminary and tentative results of analyses, (d) to record data developed in the course of an investigation, or (e) to forward information that is essentially unanalyzed and unevaluated. The review of IDA Documents is suited to their content and intended use.

The work reported in this document was conducted under contract MDA 903 89 C 0003 for the Department of Defense. The publication of this IDA Paper does not indicate endorsement by the Department of Defense, nor should the contents be construed as reflecting the official position of that Agency.

This Paper has been reviewed by IDA to assure that it meets the high standards of thoroughness, objectivity, and appropriate analytical methodology and that the results, conclusions and recommendations are properly supported by the material presented.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	January 1990	

4. TITLE AND SUBTITLE
Meta-Design -- An Approach to the Development of Design Methodologies

5. FUNDING NUMBERS
C-MDA 903 89 C 0003
TA-T-D6-553

6. AUTHOR(S)
J.E. Rogan and William E. Cralley

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Institute for Defense Analyses
1801 N. Beauregard Street
Alexandria, VA 22311

8. PERFORMING ORGANIZATION REPORT NUMBER
IDA Paper P-2152

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
OUSD(A), R&AT/ET (Dr. Leo Young)
The Pentagon, Rm 3D1089
Washington DC 20301-3080

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)* The goal of the Unified Life Cycle Engineering (ULCE) Program is to develop enhanced design environments that will allow supportability and producibility to be considered early in the design process along with the usual factors of cost, performance, and schedule. However, before such environments can be developed, a master plan or architecture for such environments must be in place. Moreover, such an architecture must be derived from the specific activities and information requirements of the design process. Thus, prior to accomplishing the ULCE program goals, a design process which addresses early consideration of producibility and supportability must be developed. Such a process must include a methodology for making design decisions and for negotiating changes in design requirements when these requirements conflict. To date, while much research has been conducted on issues related to computer integration as it applies to development of ULCE, little or no research has been done on the more fundamental question of development of ULCE design methodologies. This paper addresses a number of critical issues related to design methodology development (which is termed meta-design since it involves designing a design methodology), provides several key analytical results useful in assessing design methodologies in terms of their suitability for accomplishing ULCE, and identifies several new research directions that should be pursued if the ULCE goals are to be achieved.

4. SUBJECT TERMS Unified Life Cycle Engineering, ULCE, CAD/CAM, Meta-Design, Concurrent Engineering, Simultaneous Engineering, Design Methodology, Design Decision Support, Planning, Producibility, Supportability, Design Environments, Manufacturing, Life Cycle Cost, Acquisition, Advanced Technology

15. NUMBER OF PAGES
186

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

IDA PAPER P-2152

META-DESIGN — AN APPROACH TO THE DEVELOPMENT OF DESIGN METHODOLOGIES

J. E. Rogan, *Georgia Institute of Technology*
School of Aerospace Engineering

William E. Cralley

January 1990

INSTITUTE FOR DEFENSE ANALYSES

Contract MDA 903 89 C 0003
Task T-D6-553

PREFACE

This paper is the result of work performed by the Institute for Defense Analyses (IDA) under contract number MDA 903 89 C 0003, Task Order T-D6-553, Amendment Number 1, "Applications of Systems Engineering Techniques to the Development of a Unified Life Cycle Engineering (ULCE) Environment." This work was performed for the Air Force Human Resources Laboratory, Logistics and Human Factors Division, and the Under Secretary of Defense for Acquisition (USD(A)). The document satisfies subtask 5, which requires identification of techniques to assist a design team in a hierarchical design process in balancing conflicting design goals and requirements which have been allocated to the team either by a customer or by another higher level design team.

This paper was reviewed by Dr. Jeffrey Grotte of IDA, Dr. Joel Tumarkin, an IDA consultant, and by Dr. Daniel P. Schrage of the Georgia Institute of Technology.

ACKNOWLEDGEMENTS

The authors are indebted to Dr. Daniel P. Schrage and Dr. James I. Craig of the School of Aerospace Engineering, Georgia Institute of Technology, for their valuable comments and critique. We owe special gratitude to Dr. Jaroslaw Sobieski of NASA Langley Research Center, who assembled an editorial board at NASA to review an early draft of the paper. Their detailed comments were particularly helpful. Capt. Raymond Hill of the US Air Force Human Resources Laboratory at Wright-Patterson Air Force Base also provided a very cogent analysis of the strengths and weaknesses of that draft. Dr. Joel Tumarkin provided useful suggestions for reorganization of the paper. Dr. Daniel E. Whitney of the Charles Stark Draper Laboratory provided valuable detailed criticism of the water storage tank example. Any errors or oversights remaining in the published text are the sole responsibility of the authors.

The views and opinions expressed in this paper are the views and opinions of the authors and do not necessarily reflect the views and opinions of the United States Air Force, the Air Force Human Resources Laboratory (AFHRL) or the Department of Defense (DoD). This report reflects independent research, funded by AFHRL in support of the Air Force Systems Command initiative, Unified Life Cycle Engineering (ULCE).

CONTENTS

FIGURES

APPENDICES

ACRONYMS

CAD/CAM	Computer-Aided Design/Computer-Aided Manufacturing
CAE/CAD	Computer-Aided Design/Computer-Aided Engineering
CALS	Computer-Aided Acquisition and Logistics Support
CE	Concurrent Engineering
DoD	Department of Defense
DSB	Defense Science Board
DSS	Design Structure System
ET&M	Engineering Theories and Models
FDS	Flexible Design System
FMS	Flexible Manufacturing System
FSD	Full-Scale Development
IDA	Institute for Defense Analyses
ISM	Interpretive Structural Modeling
KKT	Karush-Kuhn-Tucker
LCC	Life Cycle Cost
QFD	Quality Function Deployment
RAMCAD	Reliability and Maintainability in Computer-Aided Design
R&M	Reliability and Maintainability
SPC	Statistical Process Control
TAAF	Test, Analyze, And Fix
TQM	Total Quality Management
ULCE	Unified Life Cycle Engineering
VLSI	Very Large Scale Integration

EXECUTIVE SUMMARY

Unified Life Cycle Engineering (ULCE) is an Air Force Systems Command Project Forecast II research and development program whose stated goal is

> to develop, demonstrate, and transfer to application the techniques and technologies needed to provide advantageous computerized integration of the procedures dealing with designing for producibility and supportability with those dealing with designing for performance, cost, and scheduling [Ref. 1]

In 1988, the Air Force requested that the Institute for Defense Analyses (IDA) develop an architecture for a computing environment within which ULCE can be implemented. A major finding of the resulting study, *Architecture and Integration Requirements for an ULCE Design Environment* [Ref. 2.], was that the concept of meta-design, the planning of the design decision process, must play a central role in an ULCE architecture. Meta-design results in a set of design decision tasks and a procedure for executing these tasks that, if carried out, will result in a design that meets the user's requirements in a near-optimal manner, as judged by certain explicit criteria. The research reported in this paper further develops the concept of meta-design, demonstrates the importance of meta-design in achieving the ULCE program goal, and presents an analytical aid for doing meta-design that is applicable to a wide variety of design problems.

A. BACKGROUND

In recent years, the issues of poor weapon system quality, high cost, and long development lead time have received considerable attention from senior management within the Department of Defense (DoD). The design process has long been recognized as a major factor contributing to these problems.

A number of DoD initiatives have advocated improved management of the design process, with particular emphasis on early consideration of factors such as producibility and supportability. Techniques, such as Taguchi Methods, the Boothroyd-Dewhurst Design for Assembly Methodology, and Quality Function Deployment (QFD) have been proposed as solutions. A well disciplined engineering process has also been advanced as the key to obtaining products that are producible and supportable.

While techniques and procedures are valuable aids in improving design practice, developing a design that is balanced in terms of performance, cost, and *ility* characteristics can pose considerable technical challenges. Situations in which a considerable advance in terms of performance is desired and in which new, unproven technologies are being incorporated into a design, present significant technical difficulties that must be overcome before a life cycle engineering approach is feasible. New weapon system design projects usually fall into this category, and disciplined management practices alone are not sufficient to address these technical challenges.

The ULCE initiative was undertaken to address these technical issues. Ultimately, a combination of sound design management practices and technology advances in areas such as computer-aided design and computer-aided manufacturing (CAD/CAM) will result in the DoD's realizing substantial savings in ownership costs of weapon systems.

B. DESIGN PROCESSES

The goal of ULCE is to develop a design environment supportive of a design process in which producibility and supportability are given early and equal consideration with cost, performance, and schedule. An ULCE design environment cannot be developed successfully without considering the process it must support. The structure of a computing environment to support ULCE must match the needs of the designers conducting the activities that constitute the design process.

Design begins with an initial specification of a set of customer requirements. In response to these requirements, the design team generates a number of design concepts that represent potential solutions to the customer's requirements. This set of concepts is then typically narrowed to a few concepts that offer the greatest promise for meeting the customer's needs. At this point, the concepts are analyzed and evaluated to determine their feasibility; the values of key design parameters that specify a particular version or instance of the concept are also determined. This information is then passed on to the next phase of design, in which further detailing of the concept takes place.

The analysis and evaluation portion of the design process may also lead to a determination that the concept is infeasible, due to conflicts in requirements or specific features of the concept. The design team must develop a thorough understanding of the design problem through conduct of trade-off analyses. The information developed through these analyses should be presented to the customer to allow him to make an informed decision regarding modifications in the requirements.

A design methodology is a plan for conducting the analysis and evaluation portion of the design process for a particular design concept and set of requirements. To be successful, a design methodology must lead--efficiently and with a minimum number of iterations--to a realization of the design concept that appropriately balances the customer's goals and requirements.

When the nature of the design concept, requirements, and associated problems fall within the domain of existing engineering knowledge, an existing design methodology can be used to conduct the analysis and evaluation portion of the design process. When a new concept is being considered (one whose principal mode of operation is not well understood, uses new technologies, or uses old technologies in new ways), a new design methodology must be developed.

A new design methodology is also needed when the nature of the customer requirements given to the design team differs from those requirements that have been specified to the design community in the past for similar problems. For example, if the requirements historically levied on designers have been such as to allow them to avoid early consideration of producibility in the design process, then specification of new requirements to the design team in which producibility is a key factor will require a new design methodology. Because weapon system designers have not commonly considered producibility or supportability early in the design process, implementation ULCE will usually require development of new design methodologies.

In this paper, the development of a design methodology is called meta-design.

C. META-DESIGN AND THE ULCE ARCHITECTURE

An existing design process geared to producing designs optimized for performance considerations without regard to cost, schedule, producibility, or supportability is not an ULCE design process, and automating such a process will not lead to ULCE. The sequence of design decisions of the existing process must be reordered to implement ULCE, and different decisions may be required. Plans for integrating CAD/CAM tools, analysis tools, and design data bases should be directed toward executing a specific ULCE design methodology. The type of ULCE design methodology used will depend on the type of design problem being addressed.

Implementing a different computer integration scheme for each design methodology would pose a considerable burden in terms of software development, however. An alternative approach, advocated in Reference 2, entails developing a flexible design system (FDS) capable of supporting the activities of methodology development (meta-design) and methodology execution (design) for multiple design problems. Such a system would be analogous to a flexible manufacturing system in that it could be rapidly reconfigured to support production of many different designs.

Simple techniques to aid in meta-design now exist--Interpretive Structural Modeling (ISM) and the Design Structure System (DSS). Both of these techniques are based on input in the form of a graphical representation of the design concept in terms of design variables and relationships among these variables. These techniques allow identification of groups of design attributes that may be determined concurrently, and placing these groups into a sequence specifying the order in which they are to be addressed. Such a grouping and ordering is called a design decision plan.

Unfortunately, the graphical representations used as input to ISM and DSS do not contain sufficient information on the analytical relationships among the design variables to guarantee that decision plans derived through their use will result in designs that are feasible or appropriately balance customer goals. Moreover, these simple techniques do not aid the design team in developing the information needed to conduct trade-off analyses when initial requirements conflict. In such situations, an analytical approach to meta-design is needed.

D. AN ANALYTICAL APPROACH TO META-DESIGN

This paper presents an analytical approach to meta-design that involves two elements:

- A framework provided by optimization theory that allows the convergence of design methodologies to be assessed
- Specific criteria that can be used as guidelines in synthesizing design methodologies.

To develop this approach, the design problem is formulated in terms of a multi-objective mathematical optimization problem. Formulating the problem in terms of optimization is important because it provides a theoretical framework for studying alternative design decision processes. In particular, the analytical notions of convergence and rate of convergence can be introduced to provide quantitative means of evaluating the feasibility and efficiency of a design decision process.

Two results from optimization theory form the foundation for the methods presented in this paper. These results allow proof of convergence of a sequence of design decisions to a balanced, feasible design. The criteria that guarantee convergence are outlined in Chapter IV, where they are illustrated by application to a simple design problem. The underlying mathematical theory is contained in Appendix B.

Formulation of the design problem in terms of optimization theory does not imply that the problem must be solved by numerical optimization methods. The approach contained in this paper uses results from optimization theory to aid in decomposing a large design problem formulated in this way into a set of smaller problems. How these smaller problems are solved is not specified. Methods other than numerical optimization, including ad hoc approaches based on engineering judgment, could be applied to yield solutions to these problems. In fact, with large problems, solution methods other than numerical optimization will likely be essential.

The approach presented in this paper leads to a method for determining the entire family of Pareto-optimal solutions (solutions in which the value of a particular design goal cannot be improved except at the expense of reducing the value of another goal) from the solution of a finite number of single objective optimization problems and partial derivatives. This information is important in the requirements negotiation process, and this method allows such information to be developed very efficiently.

E. CONCLUSIONS

The success of techniques such as Quality Function Deployment strongly depends on the existence of a design methodology that will deliver choices for design attributes that lead to balanced, feasible designs. While such methodologies exist for consumer products (which are usually based on derivatives of existing systems), such methodologies must be invented for advanced technology systems. The lack of a systematic approach for assessing whether a given design methodology will deliver a product balancing a conflicting set of requirements has been a contributing factor to the problems encountered in transitioning these systems from the development phase to production and operation. The method presented in this paper will aid in solving these problems.

Application of this method is based on optimization theory but is independent of the use of specific numerical techniques for design optimization. Thus, this approach can be used in areas of design where numerical optimization techniques have been difficult to apply, such as design problems where rough approximations must be made in the engineering theories and models underlying the concept. In these cases, engineering judgment, perhaps based on comparison of predictions made by several approximate theories, is required to determine values for design variables.

The approach developed in this paper also promises to be useful in engineering design problems having attributes that are subject to uncertainty or random variations. In this context, the results offer the means to systematically extend methods, such as those of Taguchi, to the solution of complex design problems in the development of advanced technology systems.

F. RECOMMENDATIONS FOR FUTURE RESEARCH

The results of this paper suggest that additional research should be pursued in several areas.

First, the methods of this paper should be demonstrated and evaluated by applying them to a design problem comparable in scope to the conceptual development of an aircraft system. In particular, a life cycle approach to this problem should be formulated and executed through application of this methodology.

Research into the application of the ideas contained in this paper to product development and system management problems characterized by uncertainty or random variations in the design attributes should be also be pursued. There have been a number of successful applications of Taguchi methods to develop designs that are robust under uncertainties in manufacturing process and usage environment. The methods of Taguchi, as interpreted by Tse [Ref. 24], could be used to solve the optimization subproblems identified by the meta-design procedure of this paper, allowing the Taguchi approach to be applied to design of complex, advanced technology systems.

Another promising area of research is coupling of the meta-design approach of this paper with Quality Function Deployment (a matrix technique for translating customer needs into design requirements). This would lead to an integrated approach to the total design process--from initial systems engineering analyses through planning and execution of specific design methodologies. QFD has been proven useful in a number of consumer product development activities, and has also been used successfully as a high level planning tool. Its usefulness in complex, advanced technology developments is limited by its non-analytic approach. Coupling QFD with meta-design techniques such as those of this paper could remove this limitation.

Finally, the convergence theory developed in this paper for decomposition methods of optimization should receive further investigation. Useful results concerning the convergence of iterative decomposition methods that are not sequential are immediately accessible using the techniques developed here.

I. INTRODUCTION

Unified Life Cycle Engineering (ULCE) is an Air Force Systems Command Project Forecast II research and development program whose stated goal is

> to develop, demonstrate, and transfer to application the techniques and technologies needed to provide advantageous computerized integration of the procedures dealing with designing for producibility and supportability with those dealing with designing for performance, cost, and scheduling [Ref. 1].

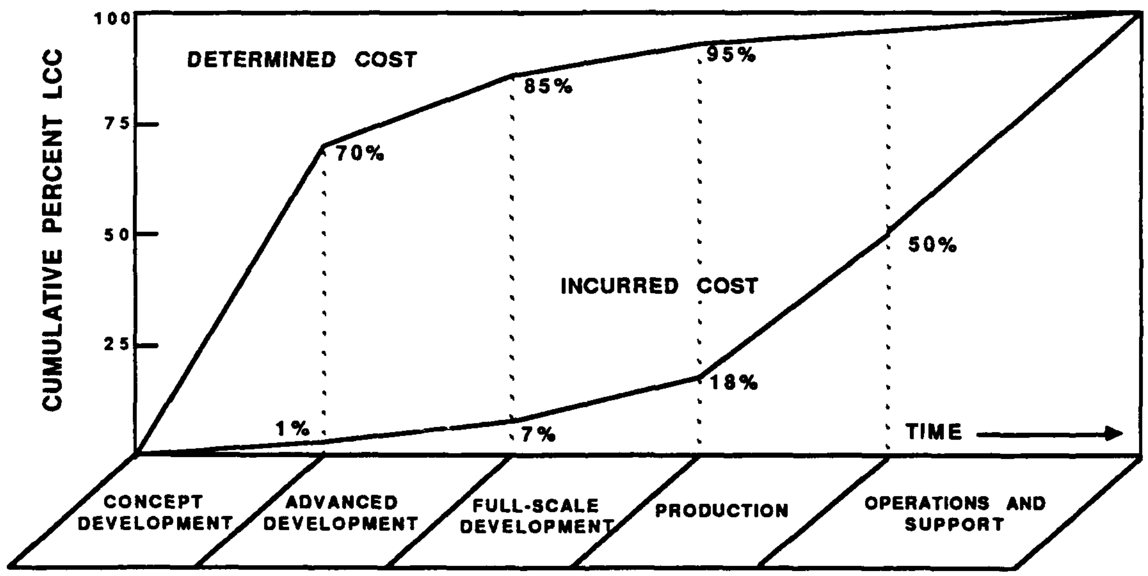
In 1988, the Air Force requested that the Institute for Defense Analyses (IDA) develop an architecture for a computing environment within which ULCE can be implemented. A major finding of the resulting study, *Architecture and Integration Requirements for an ULCE Design Environment* [Ref. 2.], was that the concept of meta-design, the planning of the design decision process, must play a central role in an ULCE architecture. Meta-design results in a set of design decision tasks and a procedure for executing these tasks that, if carried out, will result in a design that meets the user's requirements in a near-optimal manner, as judged by certain explicit criteria.

The research reported in this paper further develops the concept of meta-design, demonstrates the importance of meta-design in achieving the ULCE program goal, and presents an analytical aid for doing meta-design that is applicable to a wide variety of design problems. This research extends the work first reported in Reference 2.

A. BACKGROUND

The issues of poor quality, high cost, and long development lead time of new weapon systems have received considerable attention from senior management within the Department of Defense (DoD) [Ref. 3]. It has long been recognized that these shortcomings are due largely to the process by which weapon systems are developed--and in particular the way they are designed. For example, the Defense Science Board (DSB), in a 1982 summer study [Ref. 4], found that there was no inherent reason why high performance weapon systems utilizing advanced technologies should exhibit poor operational availability when fielded. If such systems were properly designed, with early consideration given to reliability, maintainability, and other field support factors, they would exhibit a high level of availability.

The Boeing Aerospace Company, in a study examining ballistic missile systems, found that while only 1 percent of the system life cycle cost (LCC) was expended by the end of the concept development phase, 70 percent of that system's LCC was implicitly determined by the design decisions made during the concept development phase (Figure I-1.) By the end of full-scale development (FSD), more than 95 percent of the system's LCC had been determined, although only 18 percent of this cost had been expended. Thus it is in the design phase that we have the greatest leverage over LCC--decisions made in this phase will determine most of the subsequent acquisition and ownership costs for a weapon system.



SOURCE: BOEING COMPANY

Figure I-1. Life Cycle Cost Committed versus Expended by Life Cycle Phase (Ballistic Missile System)

B. RECENT DEPARTMENT OF DEFENSE INITIATIVES ADDRESSING DESIGN

Recognizing the importance of design, during the past few years DoD has undertaken several initiatives that address, among other things, the engineering design portion of the weapon system acquisition process. The goal of these initiatives has been to improve the design process and, as a result, reap significant downstream ownership cost savings. The following initiatives are representative of the major DoD thrusts addressing design.

1. Transition from Development to Production

In a summer study conducted in 1983 [Ref. 5], a DSB Task Force examined why so many DoD weapon systems programs have experienced difficulties in making the transition from the engineering development phase to production. Serious producibility and manufacturability problems have plagued many DoD weapons programs, resulting in schedule delays, costly redesign activities, and systems with poor reliability. The DSB Task Force found that these problems could be attributed in large measure to lack of appropriate discipline in the engineering development process, especially in the design process. For example, the failure of engineering management to require that the design team consider the effect of their decisions on producibility was cited as one major factor in the transition problems.

The solution advanced by the DSB Task Force, documented in Reference 5, involves following a set of guidelines, or templates, that specify at which points in the development process certain activities should occur to minimize the risk of problems in the transition to production. Reference 6 includes similar templates. Such templates are an extension and elaboration of the standard system engineering management practices outlined in the DoD acquisition regulations and taught by the Defense Systems Management College [Ref. 7]. The first of the templates in Reference 5 addresses the most important and problematical aspect of weapon systems development--provision for adequate up-front program funding to allow a thorough design process in which downstream factors are properly considered. Without provision of such funds, successful execution of the remaining templates becomes difficult.

2. R&M 2000

The goal of the Air Force R&M 2000 Initiative is to increase the combat capability of its weapon systems by improving their reliability and maintainability (R&M) characteristics [Ref. 8]. Test, analyze, and fix (TAAF) procedures were emphasized early in the R&M 2000 Initiative. Recently, the program has emphasized incorporating R&M considerations into the design process as early as possible.

The R&M 2000 initiative advocates a design approach in which a multifunctional team uses various tools, including techniques such as Quality Function Deployment (QFD)

[Ref. 9], techniques for design for assembly such as that of Boothroyd and Dewhurst [Ref. 10], and techniques for robust design (such as those of Taguchi [Ref. 11]) to simultaneously design a product and its related processes, including manufacture and support. Such an approach is called simultaneous engineering, and is essentially equivalent to the notion of concurrent engineering, to be discussed later in this chapter.

3. Computer-Aided Acquisition and Logistics Support

The Joint Industry/DoD Task Force on Computer-Aided Logistics Support (CALS) [Ref. 12] advocated integration of R&M considerations early in the design process through use of computer-aided design/computer-aided engineering (CAD/CAE) technology. The Air Force R&D program on Reliability and Maintainability in Computer-Aided Design (RAMCAD) is presently using this approach.

In recent years, the scope of the CALS program has broadened to include the entire acquisition function (hence the change in the program's name to Computer-Aided Acquisition and Logistics Support). The program has expanded its emphasis on early consideration of R&M to include early consideration by designers of all of those *ility* characteristics, including producibility, which in their totality define the quality of a weapon system. Concurrent Engineering (CE) is the product development process advocated by the CALS program to accomplish this. Concurrent engineering addresses all of the problems that must be overcome to conduct an effective product development process.

4. Concurrent Engineering

Concurrent Engineering is defined as

> a systematic approach to the integrated, concurrent design of products and their related processes, including manufacture and support. This approach is intended to cause the developers, from the outset, to consider all elements of the product life cycle from conception through disposal, including quality, cost, schedule, and user requirements [Ref. 13].

CE thus requires the development, in parallel with the product design, of all related processes including the product's maintenance concept and logistics support structure. Clearly, CE implies the use of a multifunctional team approach to system development. This approach makes provision for the *ilities* specialists to appropriately influence the design by serving as members of the design team. The team has various tools and techniques such as QFD at its disposal to aid in structuring and tracking design activities and decisions.

Concurrent engineering emphasizes not only quality improvement but also reduction of acquisition cost and development lead time. Concurrent engineering lowers acquisition cost because errors in the design are detected and corrected early in the design process. (Errors discovered further into the development process cost more to correct than those discovered early.) Changes that can be made without modifying hardware are nearly always less costly than those requiring hardware modifications, and the amount of hardware that must be modified as a result of a design change increases as the development process progresses. Design changes made after the product is in production may necessitate retooling of the factory, an extremely expensive process.

Early detection and correction of errors also leads to reduced development lead time because errors discovered early in the development process can be addressed by fewer people than errors discovered later in the process. As the development process progresses and the level of detail of the design and its related processes increases, the number of people and organizations involved in the design effort also increases. A design change late in the process must be coordinated with all of the individuals and organizations involved, leading to serious management and communications problems--and delays in finalizing the solution to the design error.

Requirements and goals are prioritized in the design process by the order or sequence in which they are addressed by design decisions. The conventional approach to design has been: first, find a way to make the system work, then, figure out how to build the system, after that, decide out how to support the system, and finally address the problem of disposal. To accomplish concurrent engineering, we must consider at least three of these phases (design for performance, producibility, and supportability) in a tightly-coupled parallel decision-making process. The reason for doing this is to achieve a better balance between performance, producibility, supportability, cost, and schedule. Thus, the idea that design goals are balanced through the sequence in which requirements are addressed is central to the goals of concurrent engineering.

5. Total Quality Management

The goal of the DoD Total Quality Management (TQM) initiative [Ref. 14] is to promote and implement continuous process improvement throughout the DoD infrastructure. This initiative is based on the quality improvement and management methods of Deming [Ref. 15] and Juran [Ref. 16], among others. These methods stress the importance of strong leadership in an organization, use of tools and techniques for understanding processes, use of statistical process control (SPC) to track and control variability in processes, and instilling pride of workmanship and the desire for continual improvement in each member of the organization. TQM requires teamwork across functions, and development and nurturing of a team approach to improvement provides the foundation for implementing TQM.

CE can be viewed as the application of TQM principles to product development [Ref. 3]. TQM is a broader concept than CE that is also applicable to other functional areas of an organization, such as customer service and distribution. Cultural change must occur if a TQM approach is to take hold. Facilitating such change is one of the greatest management challenges facing DoD and US industry during the coming years.

C. UNIFIED LIFE CYCLE ENGINEERING

The initiatives cited in the preceding paragraphs all have goals, which if achieved, will be beneficial to DoD. However, little progress has been made in achieving these goals in DoD weapon systems acquisition. Serious obstacles must be overcome to achieve these goals, such as cultural barriers and developmental funding profiles, which are not conducive to improved design processes, and DoD acquisition regulations that sometimes impede rather than encourage better design practice. Beyond these problems, another overriding issue must be addressed if improved design processes are to be achieved--

determining a specific and detailed procedure for implementing the recommendations of these initiatives in a specific product development program.

All of the initiatives have adequately stated the problem faced by developers attempting to incorporate life cycle considerations early in the design process. Statements of the problem of life cycle engineering have been available for years (see, for example, References 17 and 18.) Stating the problem is one thing--solving it is another matter. Significant technical problems must be addressed to arrive at a solution.

1. Dealing with Complexity--The Need for ULCE

In most cases of DoD weapon system acquisition, the technical barriers to implementation of life cycle engineering can be attributed to the complexity of the system being developed, the demands being placed on the system to significantly advance the current state of the art in terms of performance, and the increase in the complexity of the design problem if life cycle considerations are introduced. A weapon system developer attempting to take a life cycle engineering approach faces an enormous task. (See Reference 19.) The team approach to product design has limitations when the size of the project becomes very large. (See Reference 20.) Management methods alone are unlikely to be sufficient to make life cycle engineering feasible for complex products.

The Air Force ULCE R&D program has advocated using the power of the computer as a mechanism for getting a handle on the complexity of the life cycle engineering problem. Computer-aided design and engineering (CAD/CAE) systems have greatly increased engineering productivity, especially in areas such as design of very large-scale integrated (VLSI) circuits and complex avionics systems. In addition, a number of standalone analysis programs are available for assessing designs for various aspects of supportability and estimating life cycle costs. The hypothesis underlying ULCE is that by integrating such programs with the designer's CAD/CAE systems, the power needed to handle the increased complexity of life cycle engineering will be made available to the design team. The proponents of ULCE believe that through computer power, life cycle engineering will become feasible, even for complex weapon systems.

2. ULCE Program Challenges

ULCE seeks to develop a computer-based environment to support the activities in a design process. As a result, development of an ULCE environment cannot be undertaken independent of this process. But, how does one specify a design process that implements the goals of ULCE? Is there an architecture for a life cycle engineering design process that can be developed and used as a basis for developing an ULCE design environment?

IDA was tasked by the Air Force to develop an architecture for an ULCE design process and a computing environment to support such a process, for a specific design problem [Ref. 2]. IDA undertook this task with a major defense contractor, Lockheed Aeronautical Systems Company. The design of a high sink rate landing gear for the C-130 was selected as the design problem to be used as a baseline for developing the architecture.

During the course of the study the study team documented the current design process for landing gear at Lockheed and arrived at the following conclusions:

- No unique ULCE design process exists, even for a specific class of design problems such as landing gear design.
- An ULCE design process will depend on the specific requirements being placed on the design (in particular, the requirements relating to producibility, supportability and performance, cost and schedule).
- An ULCE design process will also depend on the specific product being designed as well as the specific company in which the design activities are being conducted.

Thus, the problem of developing a single generic ULCE architecture, for a process or an environment, is indeterminate--no such architecture can be specified a priori. At best one can define a higher level architecture in which a key element is development of the specific design process to suit the problem at hand. If a single design environment is to be developed to support ULCE, it must be sufficiently flexible to support multiple design processes. The activity of developing a specific design process, given the design requirements and a design concept, is called meta-design. The remainder of this report details this concept.

D. ORGANIZATION OF THE REPORT

Chapter II discusses the need for meta-design by describing the nature of engineering design and the engineering design process, introduces the concept of meta-design as a planning step within the design process, and demonstrates how meta-design relates to ULCE and to the ULCE architecture presented in Reference 2.

Chapter III further refines the concept of meta-design, addressing its input requirements and outputs. This chapter also compares various approaches for doing meta-design and identifies specific requirements for a meta-design approach for design problems, such as aircraft design, that are inherently complex and exhibit tight coupling among different design disciplines. These requirements led to the development of the analytical techniques to aid in meta-design that are presented in this paper.

Chapter IV presents an analytical aid for meta-design that can be applied to a wide variety of design problems, including aircraft conceptual design. The theory underlying this approach is illustrated by applying it to a simple design problem.

Chapter V presents the conclusions of the study and outlines additional research needed to implement the algorithm in a computer based tool that can be used in real-world design projects.

Appendix A outlines specific requirements for a computing environment to support meta-design. It provides elaboration of the discussion in Reference 2 regarding the notions of object-centered environments and constraint propagation--two paradigms of advanced computing technology that are needed for an efficient computer support of meta-design and for effective integration of meta-design with the other activities in an ULCE design process.

Appendix B contains the mathematical details underlying the meta-design algorithm presented in Chapter IV, including proofs of those theorems establishing its convergence.

Appendices C and D contain more detailed examples of the application of the algorithm to problems in aircraft design.

II. ENGINEERING DESIGN AND META-DESIGN

An ULCE design environment must support an ULCE design process. An engineering design process is more than the creation of a set of engineering drawings (or computer files). A design process is a human activity characterized by creativity, conflict, negotiation, and compromise. The ULCE program, if it is to be successful, must develop an environment that supports all of these activities.

In this section, a general model of an engineering design process is presented to highlight issues that must be addressed in developing an ULCE environment and to provide a foundation for the work described in the remainder of the paper. This section places meta-design, the development of a design decision making process, in the context of the overall design process.

A. ENGINEERING DESIGN PROCESSES

For the purposes of this report, an engineering design process may be defined as a sequence of interrelated activities that begins with provision of an initial set of customer requirements and ends with one of the following:

- A complete description of a product satisfying these requirements
- A complete description of a product satisfying a set of suitably modified requirements (through mutual agreement between design team and customer)
- A determination that no product satisfying the stated requirements is feasible and that modification of these requirements is not acceptable to the customer.

Figure II-1 illustrates one view of the engineering design process as performed at one level of design detail.

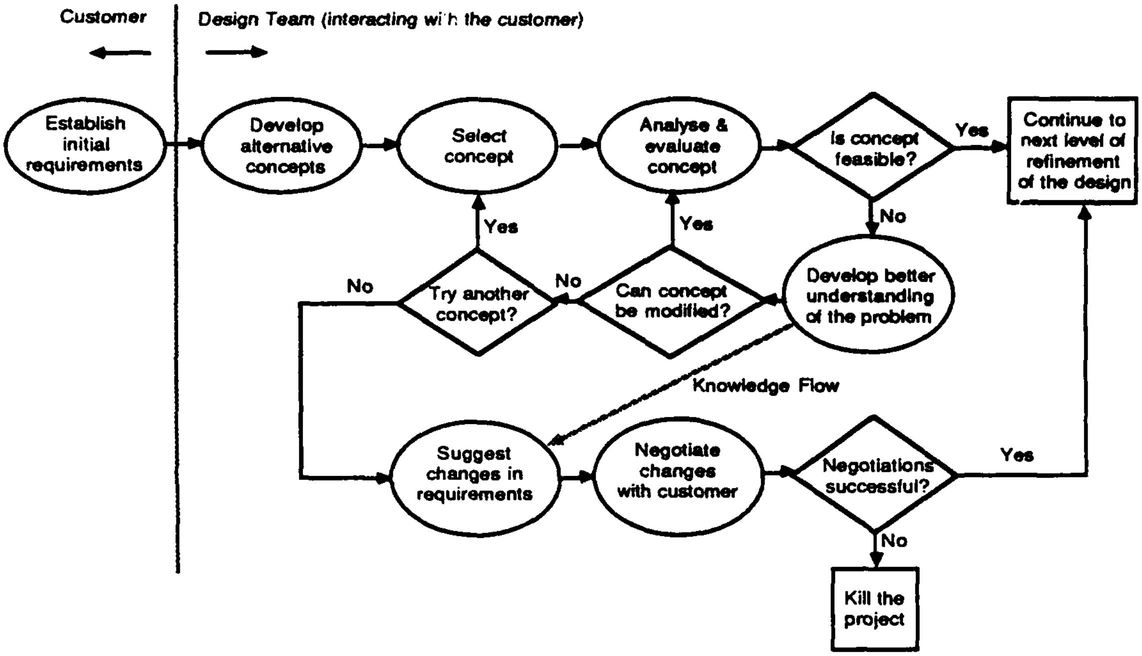


Figure II-1. Top-Level Design Process

1. Identifying and Selecting Design Concepts

The first task of the design team is to identify a large number of design concepts that could possibly satisfy the customer's requirements. The concepts are identified through brainstorming sessions or some other group process in which alternative concepts can be advanced by any member of the team. Because creativity is the key factor in developing good design concepts, the nature of computer support that is appropriate at this stage of the design process will likely be different than that provided to support later stages of design, which are more analytically oriented.

If the design team seeks to take a life cycle engineering approach, then team members who are specialists in the various *ilities* must contribute ideas. These ideas may later be determined infeasible, but all possible alternatives should be expressed and discussed at this stage of design. Selectively adopting features from one or more of these infeasible concepts may lead to a final concept with better downstream design characteristics (such as producibility or supportability).

Through the concept selection stage, feasibility of the concepts is not known--the concepts are considered possible solutions awaiting further analysis and refinement. The next step, analysis and evaluation, has as its goal establishing feasibility of a concept. Because this step is usually time consuming and expensive, the design team tends to restrict evaluations to the most promising design concepts. In this discussion, we will assume that one preferred concept is chosen for analysis and evaluation. The team could also form several subteams to analyze and evaluate several concepts in parallel. In the case of subteams, the top-level decision process would have to be altered to add a stage for selecting from among the alternative concepts after each had been evaluated.

2. Analysis and Evaluation

Only through the analysis and evaluation process can the design team determine whether the selected concept represents a viable solution to the customer's problem. The nature of the evaluation and analysis to be conducted will depend on the particular design problem. In some cases, detailed mathematical models will be created and exercised to predict the potential performance of the concept. Engineering judgment may play a key role, and prototypes may be built. The goal of all these activities is to determine--with a high degree of confidence--that the given concept can be physically realized to meet the customer's requirements.

The process by which the feasibility of a concept is determined is known as sizing. This process may involve exercising fairly elaborate computer codes that involve mathematical optimization, as is done in aircraft design. This process should establish both feasibility and a preferred configuration for the concept (specification of actual values for various parameters defining the concept, such as wingspan and weight). If the team determines that there is a good probability that the concept can be further refined to a complete design meeting the customer's needs, the process proceeds to the next level of design detail.

The sizing process can also show that the concept is not feasible--that no values can be assigned to the parameters defining the concept that will result in a design meeting the customer's requirements. It is not sufficient, however, for the design team to determine that a concept is infeasible--the team must also understand why it is infeasible and what changes can be made in the concept to make it feasible. This step involves identifying which customer requirements lead to infeasibility. Trade studies may be conducted to determine how relaxing certain requirements affects the ability of the concept to meet other requirements.

The design team should also seek to understand which features of the concept contribute to its failure. They should determine whether one or more of these features can be modified to obtain a concept that is feasible. If so, another iteration of analysis and evaluation with the modified concept should be undertaken. If this iteration establishes feasibility of the modified concept, the team proceeds to the next level of design detail (or the concept is handed off to another team for further refinement.)

3. Conflict and Negotiation

If a concept is deemed infeasible and no simple modification of the chosen concept can be shown to be feasible, the design team must decide whether to select another concept and repeat the evaluation process or to present the customer with suggested modifications of the requirements that will make the current concept feasible. Selecting a new concept and repeating the sizing process entails additional cost and design time--the decision to repeat the process will therefore depend on the project budget and schedule and the likelihood that an alternative concept will prove feasible.

If available time and funds do not permit evaluation and analysis of another concept, the design team should be prepared to present the customer with potential modifications to specific requirements that will allow the current concept to be refined into a feasible design. At this point, the information developed in trade-off analyses is crucial in aiding the design team in making recommendations and assisting the customer in making informed decisions. Should the customer accept an appropriate modification in requirements, the concept can then be sized and passed to the next stage of the design process for more detailed refinement. If the customer is not agreeable to changing the requirements or providing additional funding for exploration of additional concepts, the only alternative course of action is project cancellation.

Conflicting requirements quite frequently lead to an initial concept being infeasible--especially in weapon system developments in which a significant advance in the state of the art in terms of performance is desired (see Reference 21 for further discussion). If a design environment is to be used in support of weapon system developments, provision should be made for the environment to support the management and resolution of such conflicts.

Moreover, in contrast to mathematical optimization, in which an algorithmic solution is sought to a problem that is unambiguously stated and well understood, design problems are often stated ambiguously and are not well understood by the design team or the customer--at least at the outset of the design project. A critical element of design is the learning process that takes place among design team members (and the customer) as conflicting requirements are identified and the nature of the problem is clarified [Refs.

22,23]. Development of a deeper level of understanding of the design problem by the design team is as important a product of the design process as the final drawings for the design itself.

The distinction between design and mathematical optimization is relevant to the ULCE program, because one of the goals of ULCE is to develop a design environment that allows designs to be optimized for producibility and supportability as well as performance, cost, and schedule [Ref. 1]. Mathematical optimization is certainly one tool that can be applied to this problem in some cases. An ULCE architecture should be structured to support the total design process and not just a particular tool that might be used in that process.

4. Refinement of the Concept

The representation of the design process in Figure II-1 captures the activities at one level of refinement of the design. As the design progresses from conceptual through preliminary design and on to the detailed design phase, the results of each phase flow down to the next design phase. This arrangement of design activities is illustrated in Figure II-2.

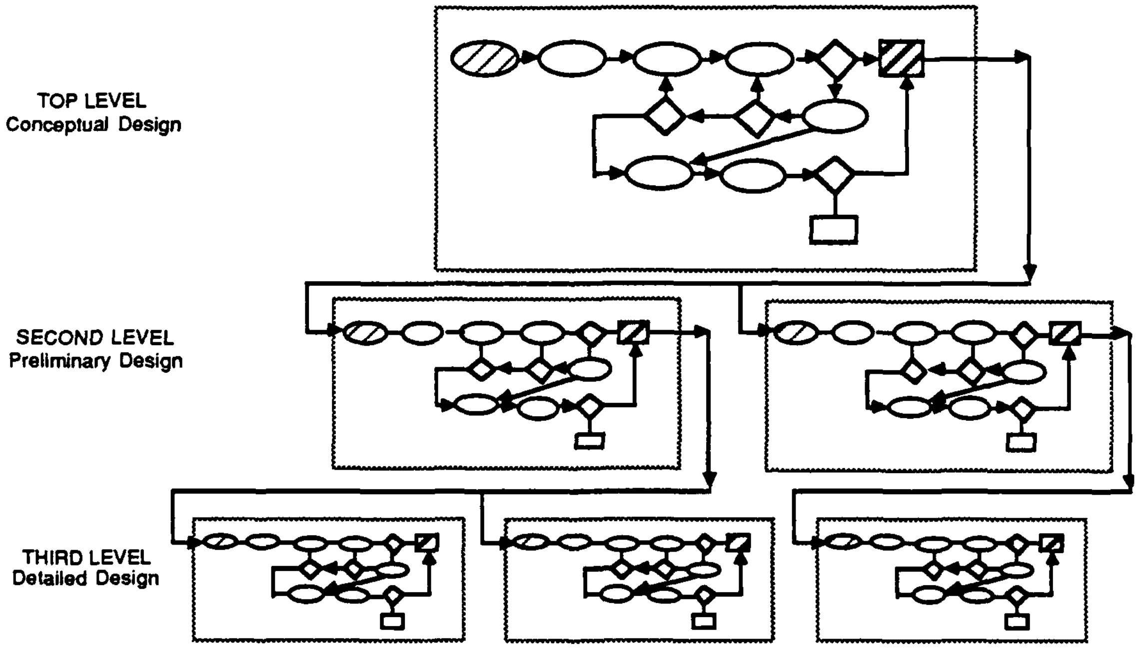


Figure II-2. Hierarchy of Design Processes

This flow-down process has associated risks--there are no guarantees that design teams at lower levels can meet the requirements allocated by the preceding level of design, and the risks are significantly greater when a new technology must be developed for a lower level design team to meet requirements. Should the requirements at a lower level of design not prove achievable, then a negotiation process must be undertaken between the lower level team and the higher level team assigning the requirements. The situation is further complicated by the fact that a change in requirements that have been assigned to one team may necessitate changes to requirements previously assigned to other teams, which results in multiple redesign activities and leads to considerable problems in coordination of efforts. This situation often results in cost overruns and schedule slippage. An analytical framework for managing these types of risk is given in Reference 24.

B. DESIGN METHODOLOGIES

A specific procedure for establishing the feasibility of a concept and developing understanding within the design team of how and why the concept works (or doesn't work) is called a design methodology. A design methodology is specific to a particular concept and set of requirements. The portion of the overall design process that constitutes execution of a design methodology is highlighted in Figure II-3.

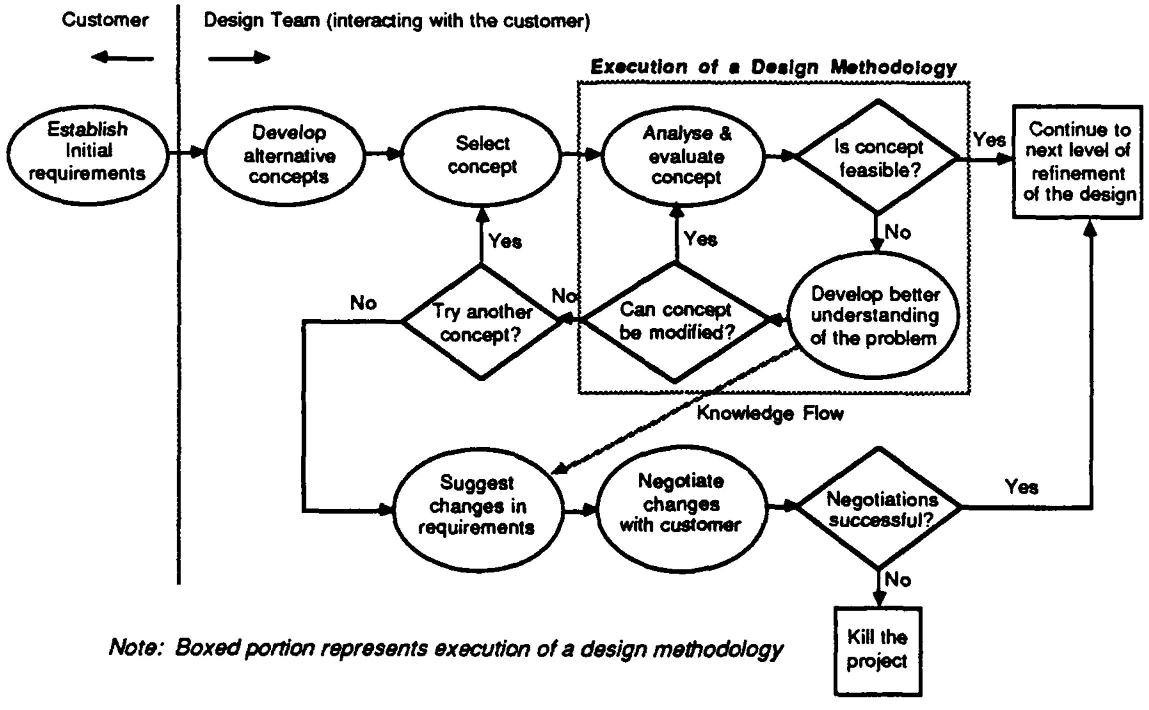


Figure II-3. Top-Level Design Process

A design methodology is a procedure for

- Determining what design decisions should be made and in what sequence
- Determining what analyses should be done to support these decisions
- Identifying specific trade studies that should be conducted.

Execution of a design methodology provides answers to the following questions:

- Can the general design concept be realized in such a way that it will meet the customer's requirements? (establishing feasibility)
- If the concept cannot be realized, what are the limiting factors or constraints that prevent a feasible realization of the concept? (generating understanding)

A design methodology is an essential component supporting the design process--it provides the means for verifying that a concept is viable and for making design decisions that will be forwarded as inputs to the next level of design. Execution of a design methodology also costs money and takes time--making efficiency an important criteria in the choice of a design methodology.

A good design methodology should also have the following characteristics:

- It should be workable--if the design concept is viable, the design methodology should lead to a feasible realization of that concept.
- It should be transparent--the methodology should be readily understood by the design team.
- It should decrease risk--by increasing the design team's confidence that the resulting design will meet the customer's requirements.

Design methodologies have been developed for many types of design problems. Some methodologies are documented in engineering textbooks and other reference works, others are retained within corporations as proprietary information used in development of new products, and others are retained within the minds of highly experienced engineers.

These methodologies have been developed over the course of the years through various means:

- Through experience and learning derived from many failures and some notable successes -- aircraft design is a good example of this approach.
- Through research activities (of companies and universities). Silicon compilation--an approach to VLSI chip design developed by Carver Mead and Lynn Conway, is an example of a design methodology developed through

research. These investigations are ultimately codified into recommended design practices.

- By analytical methods. Tools being developed at institutions such as MIT [Ref. 25] and Lockheed Aeronautical Systems Company [Ref. 26] aid designers in structuring a design decision process. Such tools are of particular value in helping designers deal with problems in which a workable methodology is unknown.
- By some combination of these approaches (most existing methodologies fall into this category).

The design methodology chosen by the team will depend on the particular concept under consideration and on the nature of the requirements that must be satisfied by the design. An industry example can be used to illustrate this. Figure II-4 represents the design methodology that was in use at Lockheed Aeronautical Systems Company (Georgia) for the design of the high sink rate landing gear for the C-130 [Ref. 2].

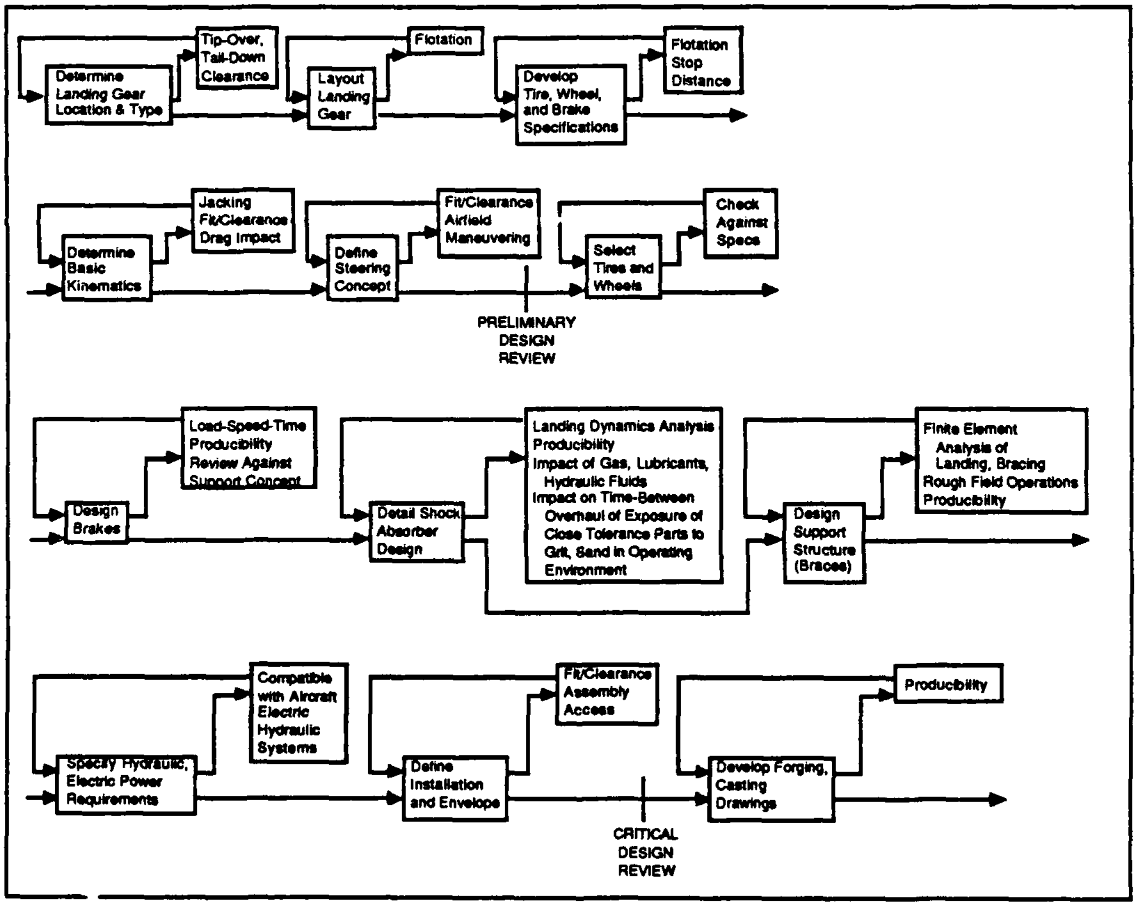


Figure II-4. Landing Gear Design Process at Lockheed

The sequence of activities and decisions in this methodology is determined largely by schedule considerations--the need to obtain a feasible design in the least amount of time (and at minimal design cost) and achieve timely delivery of design data packages to the customer. The biggest cost driver in this design methodology is the generation of detailed design data (drawings). This activity occurs late in the process, and as a result, consideration of factors such as producibility and supportability, which require detailed design information for assessment, are also pushed to the end of the process. This schedule permits very little flexibility for incorporation of changes in the design should problems be discovered relating to these factors.

If the customer specifies--as an initial requirement--that producibility and supportability were to be considered equally with schedule and cost, the methodology illustrated in Figure II-4 would not be appropriate. The sequence of design decisions would have to be re-examined and changed to accommodate these new requirements. The issue of determining the new sequence of decisions and design activities constitutes part of the activity of meta-design, which is discussed in the following section and in Chapters III and IV.

C. META-DESIGN--DESIGNING A DESIGN METHODOLOGY

When the design team has selected a promising concept that seems likely to meet the customer's requirements, the team must then plan the analysis and evaluation of the concept. They must select or develop a design methodology that they will execute to develop understanding of the concept and the requirements, prove feasibility of the concept, or identify potential changes in the concept and the requirements to recommend to the customer. This planning stage is called meta-design. During this stage, the design team is designing a portion of the design process--thus they are engaged in a design activity at a higher level of abstraction than the design of the product.

Other researchers have also defined a meta-design concept. For example, Mistree [Ref. 23] defines meta-design in two parts:

- Partitioning -- defining and partitioning a problem using a discipline independent modeling technique
- Planning -- organizing the expertise of individuals and the information (and knowledge) embodied in databases, and computers

This definition applies throughout the design process, from concept development through detailed design. However, the actual activities which make up the partitioning and planning components change as one progresses through the design process. Our concept of meta-design corresponds roughly to the partitioning portion of Mistree's definition, and we do use a discipline independent modeling technique (derived from results in mathematical optimization theory) in the analytical aid to meta-design developed in Chapter IV. Planning as defined by Mistree follows logically from the results of partitioning, but represents a step not explicitly considered in this report. Moreover, in this report, we restrict our consideration to meta-design as applied to that portion of the design process that involves the analysis, evaluation, and trade-off studies for a specific concept..

Meta-design is usually done by experienced engineers--system engineers and program managers--for the top-level design activities of a large project. However, at lower levels of the hierarchy in large design projects, meta-design is done by the design team. At the lowest level, meta-design is done by a single designer (sometimes implicitly or subconsciously) in planning his own work. Meta-design is closely related to engineering management--the choice of a design methodology will clearly affect development cost and schedule. Meta-design is the intersection of the technical and management domains of a new product development project.

1. Types of Meta-Design

Like ordinary product design, meta-design can be categorized into three classes (see Mistree, [Ref. 23]):

- *Routine meta-design*: The design concept and requirements fall within the domain of established engineering knowledge and experience. A documented design methodology is available that, if executed, will lead the team to a feasible design or demonstrate that no such design is possible.
- *Adaptive meta-design*: The design concept and requirements are beyond the bounds of current experience but appear to be similar in many respects to a situation for which a methodology is available. It appears that a nominal modification of the existing methodology may suffice for showing feasibility and developing the requisite understanding of the concept and constraints.
- *Original meta-design*: Either the concept or one or more of the requirements are of such a nature as to preclude using an existing methodology or modifying an existing methodology. A completely new methodology must be developed.

Original meta-design will be required in three situations. The first is when all of the interactions and principles underlying how the concept might work are not fully understood. Concepts utilizing advanced technologies in ways that are new, or a combination of technologies that have never been brought together before, will likely fall into this category. In such cases, meta-design may result in a decision plan in which building a prototype or many prototypes is essential to establish technical feasibility. Current analytical capabilities may not provide a satisfactory level of confidence in the performance of the design.

Original meta-design will also be required when the concept is well understood but the nature of the user requirements are beyond the domain of engineering experience. Taking a life cycle approach to the redesign of an existing fighter aircraft would be one example of such a situation--requirements involving producibility and supportability have not commonly been considered of equal importance with those involving performance for such systems in the past.

The final case requiring original meta-design is when the design team is dealing with new concepts and new requirements. In this case, a two-step approach may be taken-first finding a methodology that addresses the performance requirements and then refining this methodology to handle new requirements such as producibility. However, in some cases, new concepts and new requirements must be handled simultaneously. For example, in dealing with advanced composite materials, performance and producibility must be considered simultaneously.

2. The Relationship between Meta-Design and Unified Life Cycle Engineering

ULCE, which emphasizes consideration of producibility and supportability early in the design process, will probably require original meta-design activities. An ULCE design process for weapon systems whose technologies and concepts are new will require original meta-design, and it is also likely that original meta-design will be needed for redesign efforts of existing systems in which early consideration of producibility and supportability in the design process is beyond the scope of experience in the weapon system design community.

The relationship of meta-design to ULCE is illustrated in Figure II-5. This figure shows a number of components that must come together if an ULCE design process is to be realized. First, if the downstream *ilities* are to be incorporated into the process, a means for measuring and quantifying them must be developed. The requirements for these *ilities* must be defined in a way that is consistent with higher level customer requirements and commensurable with the requirements for performance, cost, and schedule. An appropriate description of the design concept is also needed. This description, along with the requirements, represent the required input for meta-design.

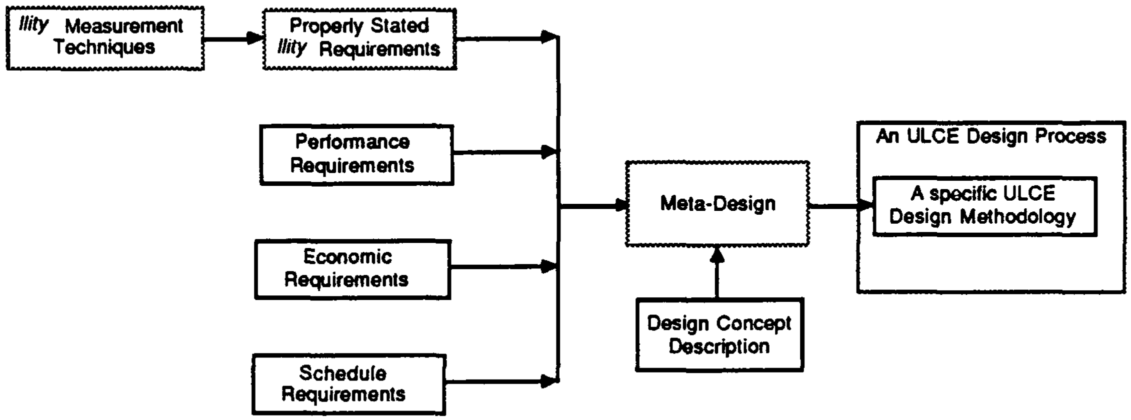


Figure II-5. Relationship of Meta-Design to ULCE

Before a computing environment to support an ULCE design process can be developed, the key components shown in Figure II-5 must exist. In particular, we must have an ULCE design methodology. A computer integration scheme based on an existing, non-ULCE design methodology may support the information transfers needed for such a methodology quite well. However, such a scheme will quite possibly be unsuited for an ULCE design methodology, in which different information transfers are necessary due to a different order of design decisions needed for ULCE.

A research and development (R&D) program seeking to develop an ULCE or concurrent engineering design environment by first modeling the current engineering design process and then developing the integrating software based on such a model is not likely to succeed, unless the current design process is an ULCE or concurrent engineering design process. If the current design process (as defined by its sequence of design activities and data flows) does not facilitate early consideration of producibility or supportability, an automated version of this process is not likely to. Data modeling efforts may lead to some understanding of an existing design process but are unlikely to be of much help in development of a new design process. A new design process must be created through original meta-design, which requires understanding the fundamental engineering principles that are driving the design problem. Once such a process is created, it can certainly be represented, at some level of abstraction, by a model of the data or information flows that must take place when the process is executed. Such a model will be an essential building block in the development of a computing environment to support execution of the process.

Thus, data and information modeling, while important tools in building computer based design environments, are not adequate in themselves as ULCE development tools. Fundamental research in development of new design methodologies is also required.

D. META DESIGN AND THE ULCE ARCHITECTURE

No unique ULCE architecture, for an ULCE design process or for an ULCE environment to support that process, exists. An ULCE design methodology will be directly tied to the design concept being evaluated and to the specific requirements being placed on the design. It will also be specific to the company or organization in which it is implemented and to the technologies used to implement it.

As a result, an ULCE design support system capable of supporting multiple design projects must provide for an explicit meta-design capability and must be flexible. In the top-level ULCE procedural architecture developed in Reference 2 (see Figure II-6), the meta-design capability is represented by the middle box, the plan design decision process stage. The relationship between the procedural architecture, as defined in Reference 2, and our representation for a design process as illustrated in Figure II-1, is shown in Figure II-7.

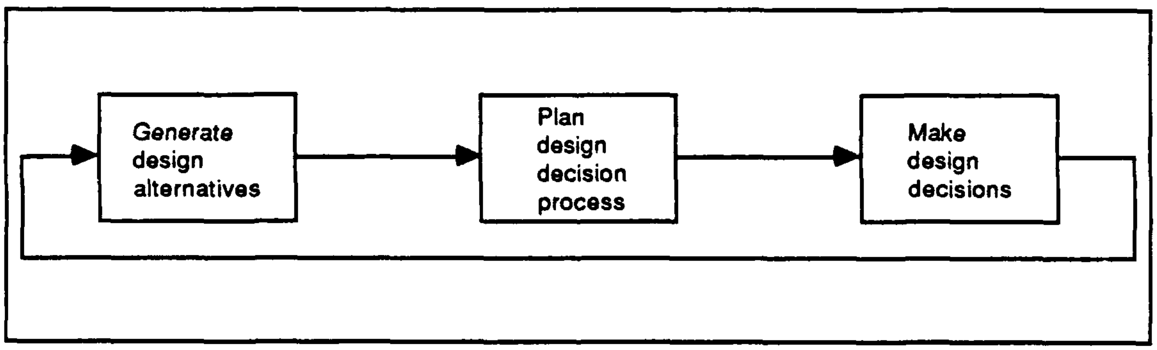
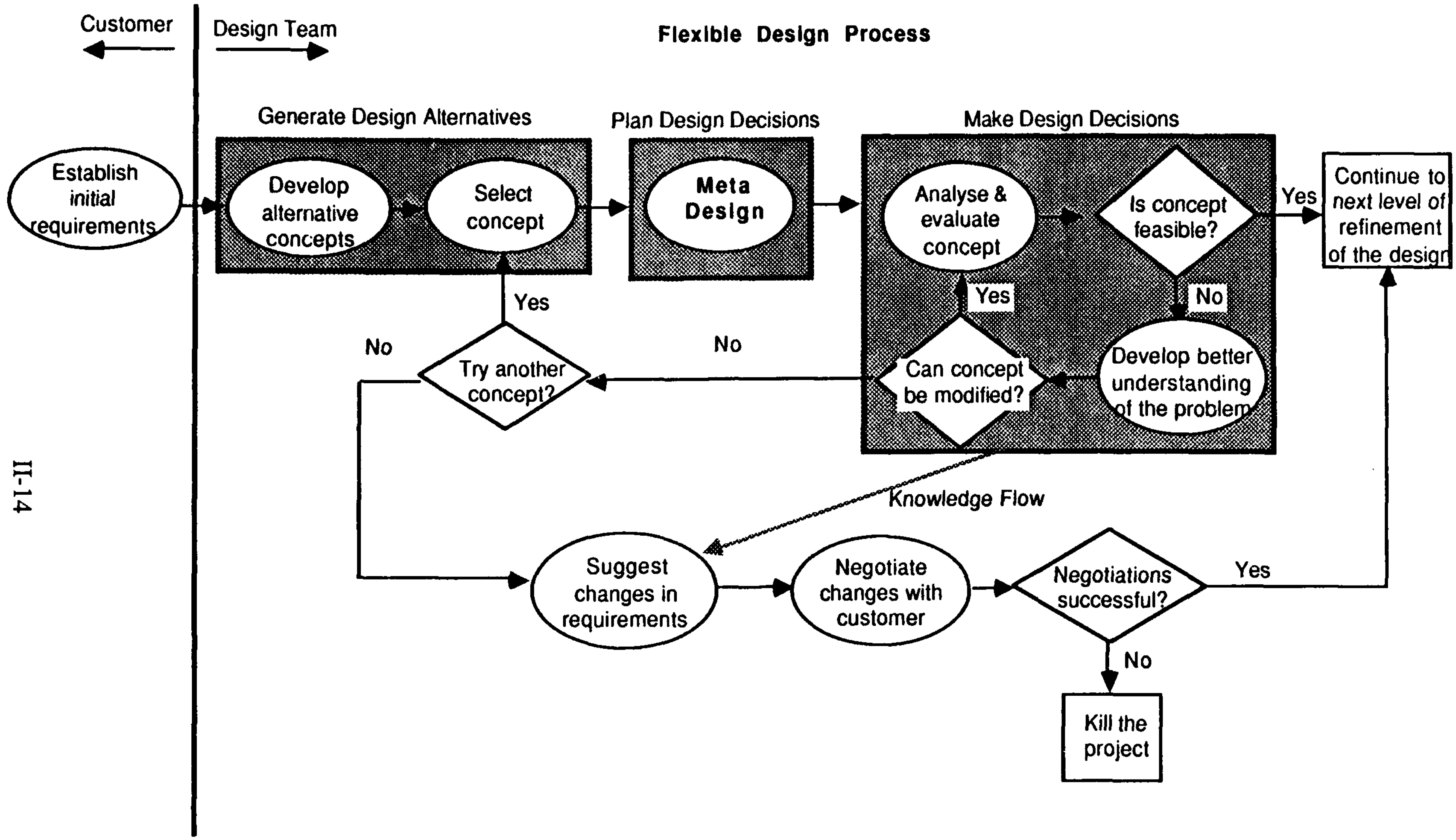


Figure II-6. Top-Level ULCE Architecture Procedural Flow



Shaded Areas Denote Major Portions of the Top Level ULCE Procedural Architecture

Figure II-7. Design Process with Explicit Meta-Design Step

The key difference between the processes shown in Figures II-1 and II-7 is the inclusion of the explicit meta-design step. If a capability for executing such a step can be devised, achieving a quantum improvement in design capabilities related to all aspects of the design (performance, cost, schedule, and downstream *ility* characteristics) is possible. As in a flexible manufacturing system (FMS), which is capable of rapid changeover and reconfiguration to make a wide variety of products, a flexible design system (FDS) such as the one envisioned under the ULCE architecture will allow generation of a wide variety of designs in much less time than is now needed.

To make such a flexible design system possible, a meta-design capability is needed. While no generic meta-design methodology is likely to exist (just as a generic methodology for ordinary design probably does not exist), analytical tools to support meta-design activities for a wide class of design problems can be developed. The following chapter discusses the input and output of meta design in more detail and shows through a simple example that an analytical approach to meta-design is necessary. A particular analytical approach that could be implemented in a meta-design aiding tool is presented in Chapter IV. This approach can be used to support meta-design in the conceptual phases of design disciplines such as aircraft design and mechanical design and can probably be also applied to problems in civil engineering and process engineering (such as chemical engineering and bioengineering).

III. REQUIREMENTS FOR META-DESIGN

Meta-design is the development of a design decision-making plan (a design methodology) based on a set of user requirements and a particular design concept. Meta-design is delimited by the information required to begin the decision planning process and the desired outputs or results. This chapter defines the input and output of meta-design and illustrates them through a simple example. Two simple techniques to aid the meta-design process, Interpretive Structural Modeling (ISM) and the Design Structure System (DSS), are presented and applied in the example. These simple approaches are based on a subset of the information required for meta-design, and their limitations are discussed. The chapter concludes by identifying the need for a more sophisticated approach such as the one presented in Chapter IV.

A. INPUT REQUIREMENTS FOR META-DESIGN

Before beginning the meta-design process, a specific design concept must be identified. The design concept may be only partially specified, but it must be clear how to state the customer's requirements in terms specific to this concept (i.e. how to formulate the requirements in terms of the design variables and parameters that pertain to this concept.) This information is developed as part of the systems engineering process.

1. The Design Concept--A Life Cycle Engineering Interpretation

If a life cycle engineering approach is to be taken in a design project, then the design concept must be considered to include, in addition to the product concept, elements of the manufacturing process, operational concept, support concept, and disposal concept. The term system life cycle concept is used in this paper to denote this broader concept of a product along with its required downstream processes and support environments. Information that must be developed for the system life cycle concept includes the functions to be performed by the system, a description of the system itself, and information describing how the system actually performs these functions.

a. Identifying Required Functions to be Performed by the System

Engineering system design begins with a statement of the need for a system, a set of requirements. Consider as an example the following requirements for a water storage system:

- Capacity must not be less than 10 cubic feet.
- Length, width, and height must each be greater than zero.
- Height must be less or equal to 2 feet.
- Relative materials cost must not exceed 6.

The statement of requirements for a system must be analyzed to identify functions that must be performed by the system to meet these requirements. The description of these functions should be independent of the particular way these functions will be implemented in the actual system. In the water storage system example, the functions are not stated in a way that specifies the geometry of the solution--we could have a rectangular tank or a spherical tank. The shape of the tank is an implementation detail--an attribute of a specific design concept that fulfills the required functions.

Functions are things that must be accomplished by the system to be designed. Functions to be performed by the water storage system include

hold water
fill
drain

For the purposes of meta-design, it is useful to broaden the definition of function to include not only those things we want the system to do, but things we don't want it to do (unintended functions). For the water storage system, such unintended functions include

contaminate water	allow water to spill
corrode	allow water to freeze
allow water to leak	allow water to evaporate.

It may also be useful to include as functions things that are done to the system as well as things the system does. For example, we might want to include actions that are performed on the system during production and support, such as

build	install	inspect	
test	repair	replace	dispose.

Implementation of one function may require implementation of several subfunctions, each of which may require other subfunctions. In this case, a hierarchical decomposition of functions will result. This hierarchical decomposition can become quite extensive and complex. Developing and managing a large functional decomposition is facilitated by techniques such as QFD [Ref. 27].

b. Defining and Describing the System

Alternative system life cycle concepts can be defined once an initial functional decomposition has been developed. Such concepts are described by attributes. These attributes may be numbers, such as height in the case of a rectangular water tank, or more complex attributes, such as manufacturing processes or support concepts.

Attributes may have alternative values, each of which leads to a different instance of a concept. For numerical attributes, the set of alternative values might be a range of numbers between lower and upper limits. Choices for the values of complex attributes may involve discrete alternatives. For example, there may be two alternatives for manufacturing process: manufacturing process A, which involves filament winding or lay-up of composite fabric on a mold, curing, and inspection operations, and manufacturing process B, which involves cutting parts from sheet metal stock, forming the parts, and welding or fastening the parts to construct a subassembly.

For the water storage system, the description of the system life cycle concept might include the following attributes:

capacity	availability	producibility	supportability
schedule	life cycle cost	acquisition cost	operating cost
disposal cost	manufacturing process	support concept	disposal plan
general arrangement	height	length	width.

The capacity, costs, and length, width, and height attributes are numerical quantities, while the other attributes, such as manufacturing process, support concept, and general arrangement, are complex attributes. Specific attributes of a concept may correspond directly to a particular function to be achieved by that concept. For example, the capacity attribute for a water storage system serves to measure the degree to which the function "hold water" is performed by the system. Other attributes, commonly called design variables, are indirectly related to functions. Such attributes would be the length, width, and height attributes of a rectangular tank concept for a water storage system. These values, taken together, determine capacity. However, many combinations of values for these attributes are possible that will result in the same functionality in terms of capacity to hold water. Thus the designer has a certain amount of freedom in choosing the values of design variables. This freedom will be limited by constraints placed on the design as part of the specification of requirements by the customer.

c. Analyzing the Ability of the System to Achieve Functions

In engineering design, the manner in which a function is implemented by a design concept is described by relationships between the attributes describing the system (the design variables) and the attributes that represent the function to be achieved. Engineering analysis entails the assessment, using these relationships, of the degree to which functions are achieved and requirements met. These relationships are specified by engineering theories and models (ET&M) that describe how the system works--how particular system elements work together to accomplish a given function or group of closely related functions.

The relationships specified by ET&Ms are often specified by mathematical formulas and equations but can also be specified procedurally--by giving an algorithm by which one attribute can be computed from values for others. A computer program or analysis code is an example of the latter method of specifying an engineering theory and model. Algorithmic specification of relationships is often used when a closed form mathematical specification is not possible.

An example of a simple engineering theory and model relating an attribute for a function to several attributes describing the system, is

capacity = length x width x height

Another example, involving attributes for a function and attributes for certain subfunctions is

life cycle cost = acquisition cost + operating cost + disposal cost

The level of detail required in the system description is closely related to the level of approximation required to apply a given engineering theory or model. Merely asserting that a relationship exists among attributes, without specifying this relationship in detail, may be considered to be engineering theory at a very rough level of approximation. However, the risks associated with basing decisions on such a crude level of analysis are probably not acceptable. Additional definition of complex attributes of the life cycle concept would be needed before more detailed engineering theories and models could be applied. For example, support cost clearly depends on the support concept, and producibility depends on the manufacturing process. Thus, complex attributes for support concept and manufacturing process would be needed if accurate modeling of support cost or producibility is desired.

2. Formulation of User Requirements and Goals

A design must meet certain requirements and constraints that are derived from various factors, such as the environment in which the product will be used. These constraints must be appropriately formulated in terms of the attributes of the system life cycle concept. An example of a constraint would be a limitation on certain dimensions of the design (in the water storage system example, such a constraint would be the height limitation). To be feasible, an instance of a system life cycle concept must satisfy such constraints.

The customer may also have certain desires that are goals that the design team should seek to achieve but are not strict requirements. Design goals specify that a particular attribute is to be maximized, minimized, or close to a target value. In a situation in which a design team is developing a product to compete in the market place (such as a new automobile), a competitive strategy is implemented through specific design goals (in this case provided by the marketing group) to guide the design team.

Taguchi methods [Ref. 28], in which the design team seeks to develop a design minimizing a loss function, are one implementation of a competitive strategy in which a robust design is desired. The loss function is a measure of the loss the customer will incur as the product deviates, due to the influence of noise factors, from a set of target values of certain design attributes. Choosing values for design attributes so as to minimize the expected loss function will result in a product that is reliable in operation--an increasingly important product characteristic to consumers.

The customer often has multiple goals that he desires to achieve in the system. ULCE presents five broad goals to the design team: maximize producibility, maximize supportability, minimize cost, maximize performance, and minimize development lead time. Because simultaneously achieving multiple design goals is often not possible, the customer's ranking of requirements becomes important. Through trade-off analyses, the design team should seek to achieve a balanced design in which each of the goals is achieved to the maximum extent possible, given the restrictions posed by the other goals and the customer's relative priorities for achieving each goal.

Such a design is a member of the set of Pareto-optimal designs, that is, designs in which an improvement in the value of any specific design goal can be obtained only at the expense of lower performance with regard to one of the other goals. A key factor in achieving a balanced design is the sequence in which design decisions are made. If producibility is an important design goal, for example, then it should be considered early in the design decision process. Meta-design must provide a means by which a decision sequence leading to a suitably balanced design can be developed.

B. OUTPUT OF A META-DESIGN PROCESS

Planning a design decision-making process involves two steps:

- Identifying design decisions
- Sequencing design decisions.

Design decisions are groups of life cycle attributes that must be co-determined; these attributes are tightly coupled and thus must be dealt with as a group rather than as independent entities. Design decisions may be considered subproblems of the total design problem. The set of these decisions, along with the sequencing information, constitutes a decomposition of the total design problem. This decomposition is the output of the meta-design process.

Sequencing of design decisions should be determined by the customer's prioritization of requirements and design goals. Competitive design strategies are thus implemented through the sequence of design decisions. Design decisions made early in the process will determine values for certain attributes, which will constrain the values of other attributes to be addressed in later design decisions.

Attributes that are strongly coupled with high-priority design requirements or goals should be addressed in early design decisions to ensure that the maximum flexibility is available to the design team to meet these requirements and attain the goals. Attributes associated with lower priority requirements will generally not be addressed until later in the process, when less flexibility is available and needed, since these attributes do not affect critical design requirements.

The method to be used to determine the values of the attributes in each design decision will not be specified by the design decision plan. Although identifying the techniques used to solve the various design decision problems is an important part of meta-design, it will not be addressed in this paper. The techniques used may vary from strictly mathematical methods (such as numerical optimization), to combinations of mathematical methods, computer models, hard modeling efforts, and engineering judgment.

A key part of the design decision plan to be developed by a meta-design process is the capability to handle the requirements negotiation process. Requirements negotiation is necessary when the initial requirements, as stated by the customer, lead to infeasibility. When design requirements conflict, the design methodology must facilitate the design team's understanding of the effects of relaxing the various constraints. Moreover, the methodology must allow design trade-off analyses to be conducted in an efficient manner. The requirement for a design methodology to support the requirements negotiation process sets fairly stringent limitations on how meta-design is accomplished and the nature of the information needed to do meta-design.

C. SIMPLE APPROACHES TO META-DESIGN

Meta-design is closely related to two techniques for decision planning, ISM [Refs. 29, 30] and DSS [Ref. 32]. These techniques are based on a graphical representation of the information in the system life cycle concept description.

Various levels of detail can be present in different graphical representations of the information. In one representation, attributes as well as engineering theories and models are the nodes (or vertices) of a graph. Figure III-1 contains such a graph for the water storage system example. An edge connects an attribute to an ET&M if the attribute appears as a variable in that engineering theory/model.

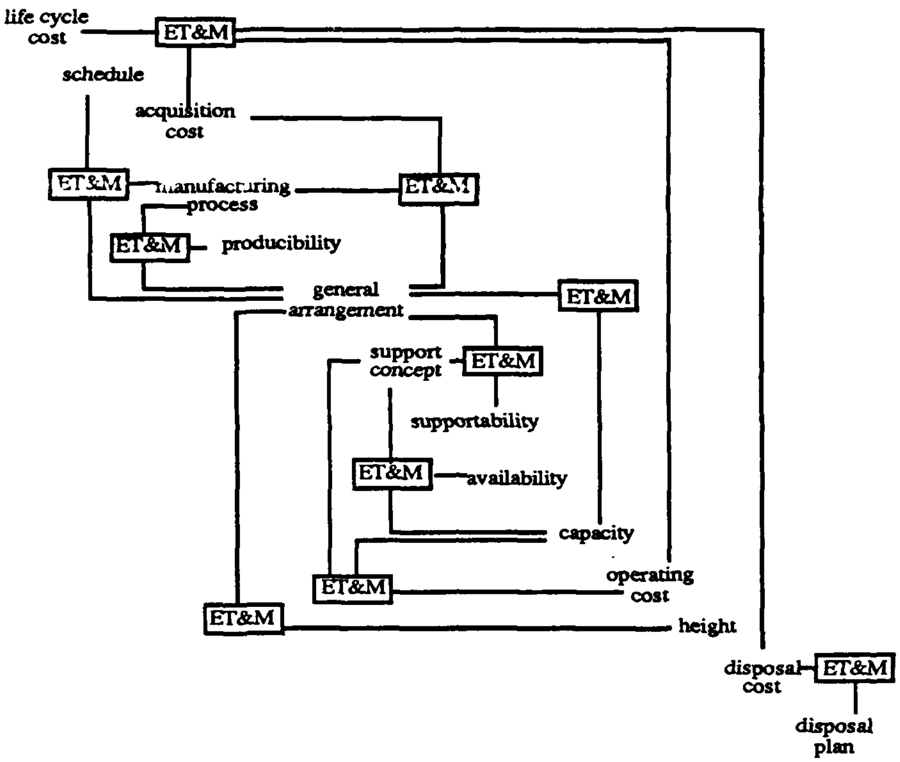


Figure III-1. Graphical Representation of the System Life Cycle Concept

Both ISM and DSS work with this information in a reduced form, a directed graph, which is obtained by eliminating the nodes corresponding to engineering theories and models. Engineering theories and models are represented indirectly in the directed graph by drawing an arrow (directed edge) from an attribute A to another attribute B. The direction of this arrow indicates that B is to be determined from A. A directed graph corresponding to the water tank example is shown in Figure III-2.

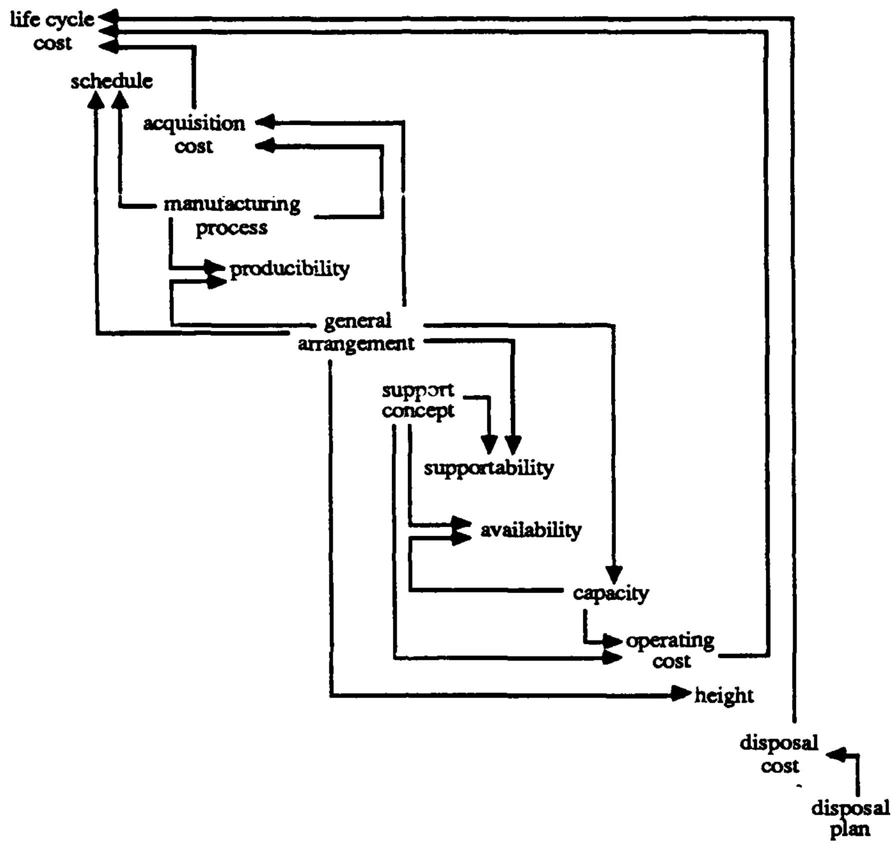


Figure III-2. Directed Graph Representation

The graphs in Figures III-1 and III-2 represent abstractions of the information contained in the system life cycle concept, in that they contain only a portion of the information in the total concept description. For example, the complete description specifies that life cycle cost is the sum of acquisition, operating, and disposal costs, while Figure III-1 indicates only that some relationship, defined by an engineering theory and model, relates these quantities; the specific nature of the relationship is not specified in the graph. Thus, a meta-design procedure based on a graphical representation will only depend on the topological structure of the system life cycle concept, not on the actual analytical details of the concept. Such a procedure may not yield a workable design decision plan, as is shown in the following paragraphs.

The directed graph obtained in Figure III-2 also depends on the directions chosen for the arrows used to represent the engineering theories and models. Assigning directions to the arrows amounts to imposing a precedence relationship on the attributes in the system life cycle concept. The implications of the need to assign such a relationship is to be addressed after the discussions of ISM and DSS that follow.

1. Interpretive Structural Modeling

ISM allows one to structure a decision-making problem based on input represented by a directed graph (the ISM problem structuring technique is described in Reference 29). To illustrate the ISM technique we shall apply it to structure the directed graph of Figure III-2.

The first step of the ISM problem structuring method is to identify all of the nodes of the directed graph that are terminal, that is, no arrows emanate from these nodes. In Figure III-2, these nodes correspond to life cycle cost, schedule, producibility, supportability, availability, and height. Once these nodes have been identified, they are placed at the top level of the restructured graph, ISM Level 1, as shown in Figure III-3.

ISM Level 1

life cycle cost	schedule	producibility	supportability	availability	height

Figure III-3. Level 1 of Interpretive Structural Modeling Approach

A new, reduced graph is then constructed by removing these nodes (and all the arrows incident on them) from the original graph. This reduced graph is shown in Figure III-4. Some other nodes will now be terminal in the reduced graph. In the example, these nodes correspond to acquisition cost, operating cost, and disposal cost. These nodes are placed at Level 2, as shown in Figure III-5. The process is iterated until no nodes remain. In the final step, arrows are added to the structured graph wherever they occurred in the original graph, as shown in Figure III-5.

ISM has been applied to design problems [Refs. 30, 31] and has been found to be useful in clarifying relationships among elements of the design problem. For our example, a design strategy might be to balance measures of producibility, life cycle cost, and supportability with availability (as a performance metric for the water storage tank) and schedule, while satisfying the requirement that the height be no greater than 2 feet. All of these considerations have found their way to ISM Level 1 in the problem structuring process. It is also clear from the structured graph of Figure III-5 that acquisition, disposal, and operating costs are intermediate quantities in that they appear at neither the lowest nor the highest level of the structured graph. Of course, we may wish to constrain intermediate quantities such as acquisition cost, and nothing in the ISM problem structuring precludes this possibility. Finally, the considerations appearing at the ISM Levels 3 and 4 tend to be the life cycle concepts, such as the manufacturing process, support concept, and general arrangement, which an implementation team can manipulate to influence the objectives and constraints at ISM Levels 1 and 2.

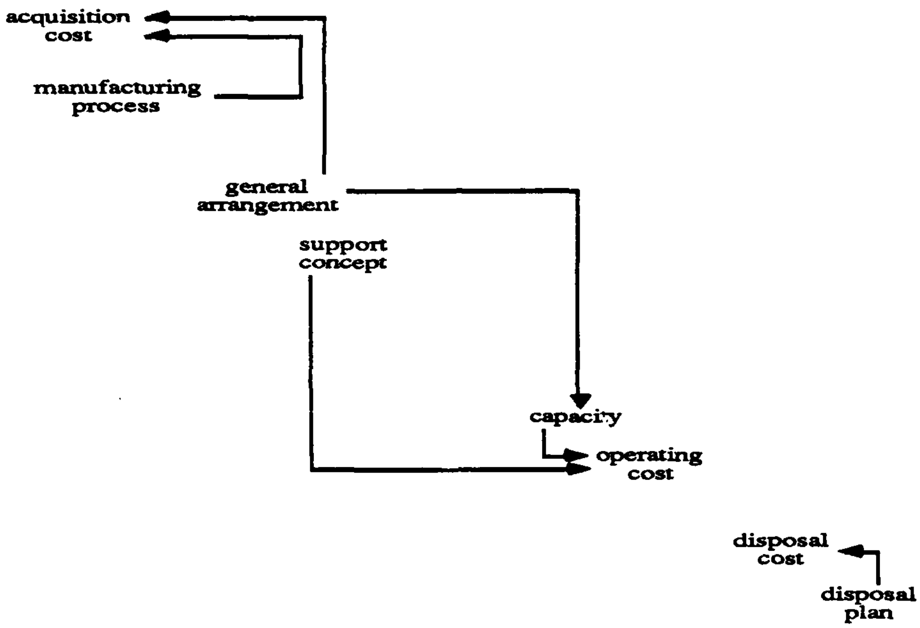


Figure III-4. Directed Graph with Level 1 Nodes and Arrows Removed

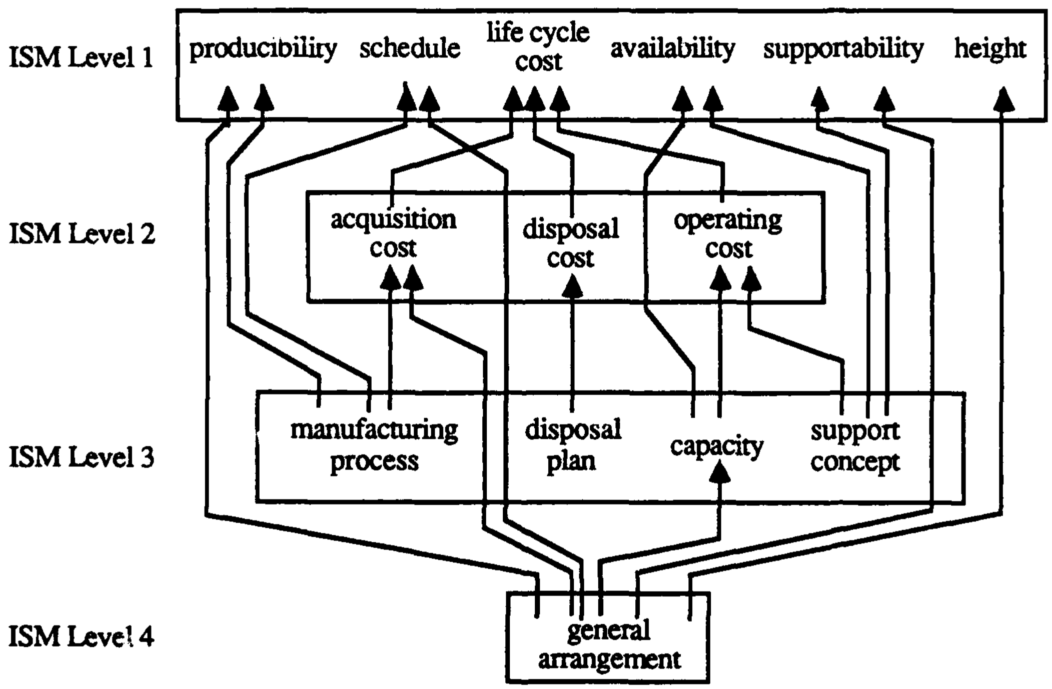


Figure III-5. Final Results of Interpretive Structural Modeling

Design decision planning results in groupings of the design variables into decision elements and sequencing of the decision elements. For example, the attributes at a given ISM level are decoupled from other attributes at the same level. If such decoupling is desired in the decision-making process, the ISM levels can be identified as design decision stages. Presumably, the sequence of decisions would follow the directions of the arrows, that is, ISM Level 4 would be solved first, and ISM Level 1 would be the last decision.

2. Design Structure System

DSS is based on the idea that feedback iterations should be reduced or eliminated in a workable decision sequence. In the DSS approach [Ref. 32], the directed graph is represented by an adjacency matrix, as shown in Figure III-6, or by an N^2 matrix [Refs. 33, 34] as shown in Figure III-7. A directed graph is represented as an adjacency matrix by indexing the nodes of the graph with integers $\{1, 2, 3, \ldots\}$. A "1" is entered in the ith row and jth column of the adjacency matrix if there is an edge in the graph directed from node i to node j. The rest of the entries in the adjacency matrix are zero. The N^2 matrix (Figure III-6) represents much of the same information in a more graphic style. The names of the nodes are entered as the diagonal elements of the N^2 matrix. An arrow directed from a diagonal element i to a diagonal element j is represented as in the adjacency matrix, except that the 1's are replaced by circles, and lines are drawn to connect the circles to the appropriate diagonal elements. The zero entries of the adjacency matrix are omitted from the N^2 matrix.

Structuring the (N^2 or adjacency) matrix of the design problem to eliminate feedback loops results in a block-diagonal structure (Figure III-8). In the simple example of Figure III-7, all feedback loops can be eliminated. In a more complex decision-making problem, some loops may be unavoidable. For such an example, a block diagonal structure can be defined on the matrix with feedback loops within only the blocks and all connections between distinct blocks strictly feeding forward. Reference 32 describes an interactive program for defining a block-diagonal structure on the adjacency matrix resulting in the reduction or elimination of feedback loops. Researchers at NASA Langley are currently experimenting with a design decision planning tool based on these ideas (Refs. 35 and 36).

	life cycle cost	schedule	acquisition cost	manufact-uring process	produc-ibility	general arrange-ment	support concept	support-ability	avail-ability	capacity	operating cost	height	disposal cost	disposal plan
life cycle cost	0	0	0	0	0	0	0	0	0	0	0	0	0	0
schedule	0	0	0	0	0	0	0	0	0	0	0	0	0	0
acquisition cost	1	0	0	0	0	0	0	0	0	0	0	0	0	0
manufact-uring process	0	1	1	0	1	0	0	0	0	0	0	0	0	0
produc-ibility	0	0	0	0	0	0	0	0	0	0	0	0	0	0
general arrange-ment	0	1	1	0	1	0	0	1	0	1	0	1	0	0
support concept	0	0	0	0	0	0	0	1	1	0	1	0	0	0
support-ability	0	0	0	0	0	0	0	0	0	0	0	0	0	0
avail-ability	0	0	0	0	0	0	0	0	0	0	0	0	0	0
capacity	0	0	0	0	0	0	0	0	1	0	1	0	0	0
operating cost	1	0	0	0	0	0	0	0	0	0	0	0	0	0
height	0	0	0	0	0	0	0	0	0	0	0	0	0	0
disposal cost	1	0	0	0	0	0	0	0	0	0	0	0	0	0
disposal plan	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Figure III-6. Adjacency Matrix of the Directed Graph of Figure III-2

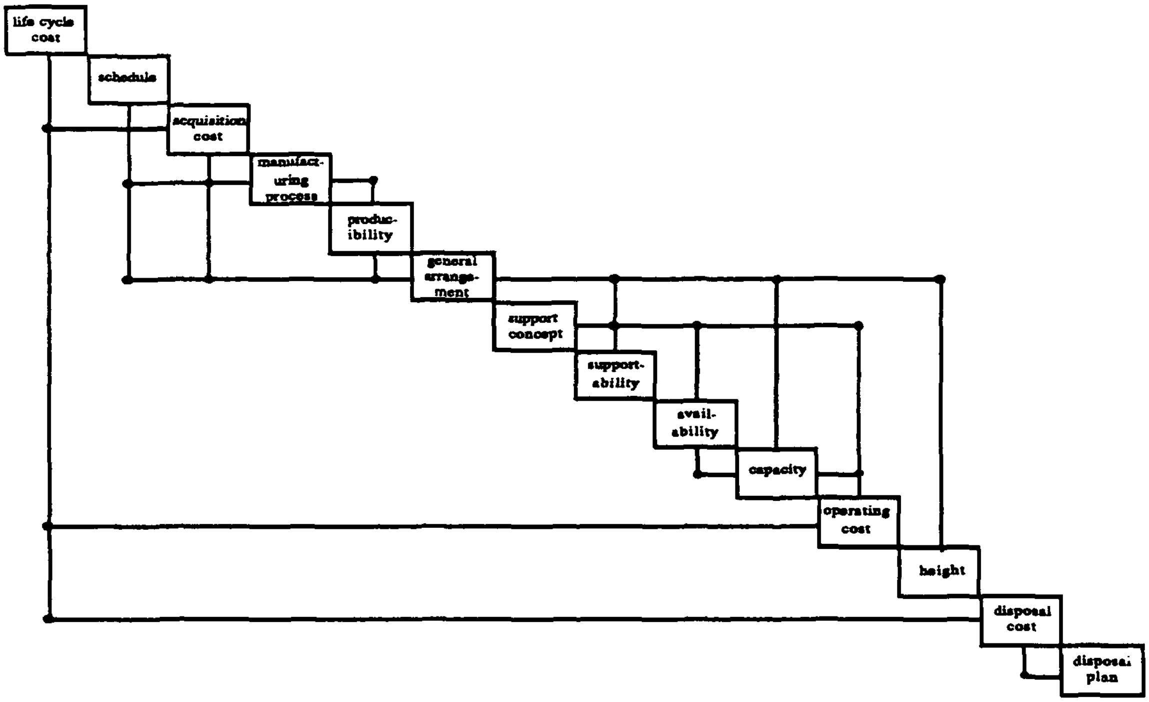


Figure III-7. N^2 Matrix of the Directed Graph of Figure III-2.

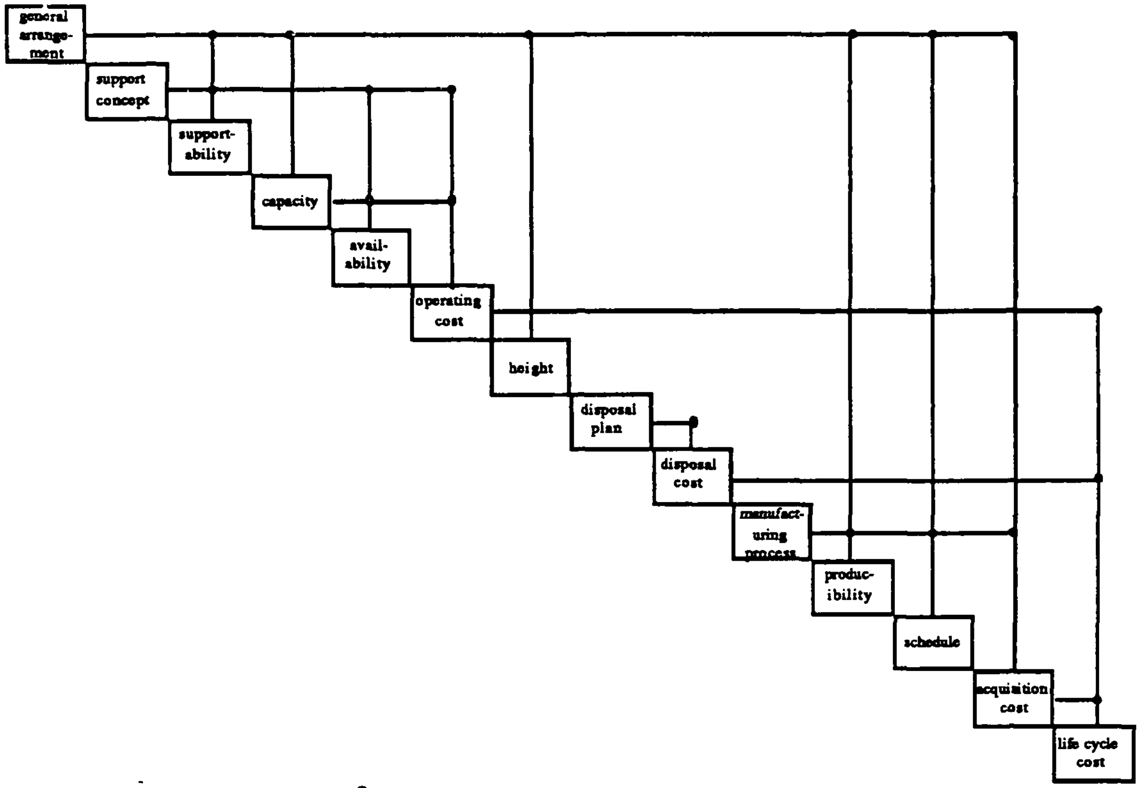


Figure III-8. N^2 Matrix Ordered Using the DSS Technique.

The blocks may be identified with design decisions. The links between blocks then specify an ordering of design decisions that may be used to establish a sequence of design decisions. Since several decision sequences without feedback loops are possible for this simple problem, any of these decision sequences are compatible with the DSS philosophy. The decision-making plan developed using this algorithm may be reviewed by the design team. Modifications to the decision-making plan can then be made by changing the directions of the arrows in the directed graph representation.

3. Limitations of Interpretive Structural Modeling and the Design Structure System for Meta-Design

As noted in the preceding paragraphs, both ISM and DSS require that a precedence relationship be established among the elements of the system life cycle concept. More than one precedence relationship can be established from these elements, yet neither ISM nor DSS tell the design team which of the resulting methodologies is to be preferred over the others.

Moreover, a design methodology obtained from ISM or DSS may not always lead to a feasible design. Should the requirements not allow a feasible design, neither ISM nor DSS provide the design team with any information about which requirements to relax to obtain feasibility or how to structure trade-off analyses to inform the customer how to modify the requirements. To illustrate the limitations of ISM and DSS for meta-design, we shall consider the following modification of the water storage system example.

In this example, we may have three attributes (length, width, and height) and two functions (capacity and relative materials cost), which are related analytically by the engineering theory and models:

[capacity] = [length] x [height] x [width]

[relative materials cost] = 2 ([length]x[width]+[length]x[height]+[width] x [height])

Assume the customer has specified the following design requirements and goals:

Requirements:

- Capacity must not be less than 10 cubic feet.
- Length, width, and height must each be greater than zero.
- Height must be less or equal to 2 feet.
- Relative materials cost must not exceed 6.

Goals: Maximize capacity, minimize relative materials cost.

Note that the design concept, functions, system attributes, and engineering theories and models have been specified in the statement of this design problem. Thus, this problem has been posed at an appropriate point in the system engineering process for the application of a design methodology. The scope of the problem is limited, representing a detail of a life cycle engineering problem, to simplify the discussion.

Figure III-9 shows the graphical relationship representing this design concept, which is analogous to Figure III-1.

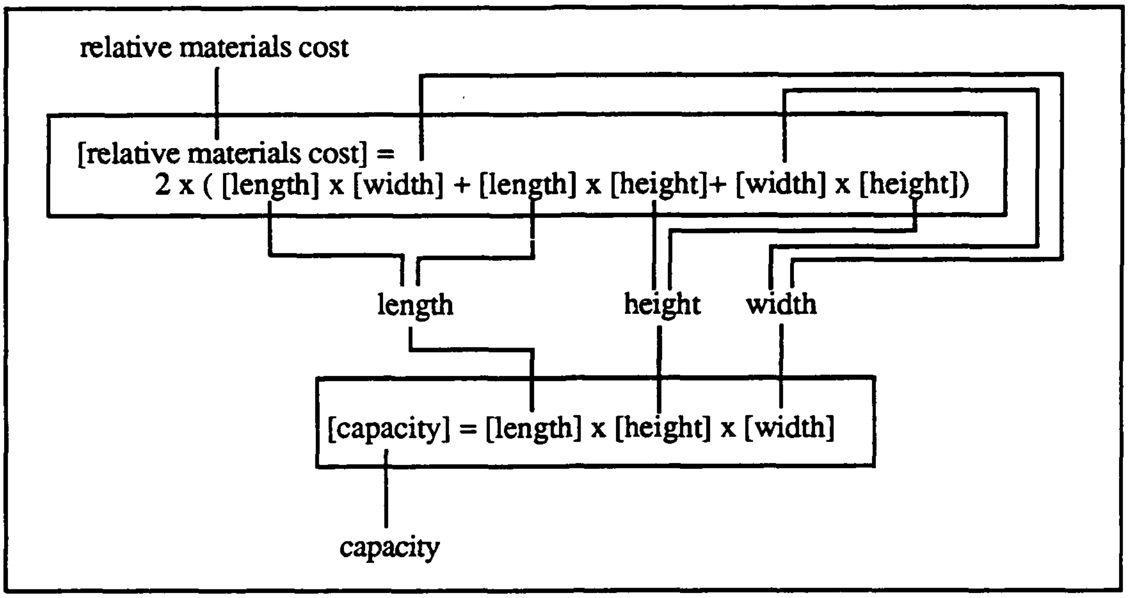


Figure III-9. Full Graphical Representation of Concept

Two alternative ways of representing this information in a directed graph are shown in Figure III-10.

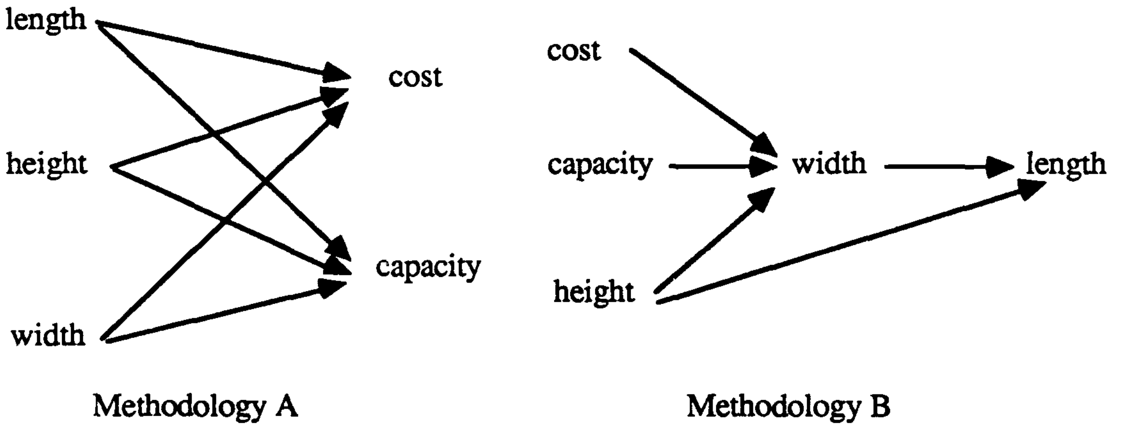


Figure III-10. Two Directed Graph Representations for Concept In Figure III-3

These two graphs correspond to two different design methodologies:

Methodology A

1. Determine length, width, and height.
2. Apply

[capacity] = [length] x [height] x [width]

to determine capacity.

3. Apply

[relative materials cost] = 2([length]x[width]+[length]x[height]+[width] x [height])

to determine relative materials cost.

Methodology B

1. Fix capacity, relative materials cost and height.

2. Solve

[length] x [height] x [width] = [capacity]

for width,

[width] = [capacity]/([height] x [length]).

Substitute this relationship into the equation

[relative materials cost] = 2([length]x[width]+[length]x[height]+[width] x [height])

and solve for length.

The two methodologies derived from the directed graphs in Figure III-9 have significant limitations. For example, from the directed graph of methodology A, we see that unless the correct values for the attributes length, width, and height are known, meeting the relative materials cost and capacity requirements without iterating the decision-making process is difficult, if not impossible. Methodology A does not specify such an iterative strategy, so undertaking this methodology would be risky.

The directed graph of methodology B indicates that we should be able to determine the design variables length and width from the requirements for cost, capacity, and height, the requirements. However, beginning with relative materials cost = 6, capacity = 10, and height = 2 (which would certainly appear to meet the design requirements), we can show that no real solution exists for width. Thus, the initial design decision setting values for cost, height, and capacity as specified in the methodology results in infeasibility in a subsequent decision.

The directed graph representation reveals no indication of this problem. In fact, the directed graph indicates that methodology B is well matched to the water tank design problem. This example shows that determining whether a design methodology will lead to feasible designs using only the information in the directed graph representation is not possible. We must retain the analytical information--originally discarded when we went to the directed graph representation--to determine whether the methodology corresponding to the graph will lead to a feasible design. Specifically, we need the analytical content of the engineering theories and models to bring feasibility into the evaluation. The connectivity information contained in the directed graph representation is simply not adequate for this task.

No feasible designs satisfy the requirements as stated by the customer in this example, which is a common situation in real design efforts. The requirements, as stated originally by the customer, conflict. In such a situation, the major issue the design team face is providing useful feedback to the customer regarding the possible trade-offs that may be made among the major goals, which, in this case, are those for capacity and relative materials cost. Neither the ISM nor the DSS approaches, based on simple directed graphs, provide the team with adequate information to do this.

The next chapter presents a method for analyzing the engineering theories and models to aid in developing a design decision-making process that does converge to a balanced, feasible design, if one exists. The method also assists the team in performing trade-off analyses in support of the requirements negotiation process in the case where the initial requirements conflict. This method provides a powerful technique for systematically evaluating the capability of a design methodology to support life cycle engineering.

IV. AN APPROACH TO THE SYNTHESIS OF DESIGN METHODOLOGIES

This chapter presents a systematic, analytically based approach for developing design methodologies. This approach has two principal elements:

- A framework provided by optimization theory that allows the convergence of design methodologies to be assessed
- Specific criteria that can be used as guidelines in synthesizing design methodologies.

Two results from optimization theory form the foundation for the methods presented in this chapter. These results allow proof of convergence of a sequence of design decisions to a balanced, feasible design. The criteria that guarantee convergence are outlined in this chapter. The underlying mathematical theory behind the results presented in this chapter is contained in Appendix B.

We go on to develop an interesting application of these ideas. The water storage tank example, introduced in Chapter III, is taken up once again. We now approach the water storage tank design problem as a problem in design methodology synthesis. We discuss alternative problem formulations, and identify the need for a "requirements negotiation subproblem".

It is not possible to meet all of the requirements imposed on the water storage tank design. Thus, the requirements, scaled and formulated as goals, become multiple objectives in a Pareto-optimization (requirements balancing) problem. In the Pareto-optimization formulation, the objective function is a weighted sum of the conflicting multiple objectives. The weighting factors can be interpreted as a relative prioritization of the conflicting requirements. A solution of the corresponding optimization problem for fixed values of the weighting factors is called a Pareto-optimal design. Pareto-optimal designs are also characterized by the statement that we cannot move closer to achieving any of the goals without moving further away from at least one of the other goals.

Since the Pareto-optimal designs are parameterized by the weighting factors or relative prioritizations, they form a manifold in the design space with each point defined by the solution of an optimization problem. Exploration of this design space can be prohibitively expensive if the solution of the optimization problems is costly. Instead, we propose a technique that allows us to determine the entire family of Pareto-optimal solutions explicitly from the solutions to a finite number of optimization problems, one corresponding to each of the conflicting objectives. This technique is applied to the requirements negotiation subproblem for the water storage tank design process.

The idea is to negotiate goals and priorities with the customer, based on what can be achieved by the optimally balanced design. The difficulty in complex design problems, such as those encountered in aerospace systems design, is that these negotiations must be carried out without explicitly solving the design problem beforehand. In dealing with a simple design problem such as the water storage tank, one's natural inclination is to find the complete solution to the problem, and then to parameterize that solution to present the information needed to support requirements negotiation. However, there is an important difference between the water storage tank example and design problems in the life cycle engineering of complex systems: closed form solutions rarely exist for complex problems. The development of the water storage tank example in this chapter has been guided by the principle that the methods used to solve this simple problem must be applicable to the solution of complex problems in life cycle engineering. The requirements must be negotiated before we seek the optimal design. This allows us to control the risks associated with investment in design development before a workable set of requirements has been established.

A. FORMULATION OF THE DESIGN PROBLEM

Once a system life cycle concept has been chosen, the design decision-making problem can be formulated as a Pareto-optimization problem:

$$\begin{aligned} \text{minimize:} \quad & \Sigma\, \omega_r\, f_r(X) \\ \text{Subject to:} \quad & \mathbf{g(X) \leq 0} \\ & \mathbf{h(X) = 0}, \end{aligned}$$

where **X** is a vector of design variables (assumed to lie in a compact subset of $\mathbf{R}^n$), $f_r(X)$ are design goals or objectives, and **g(X)** and **h(X)** are vector functions of the vector **X** that represent requirements or constraints. The ω_r are relative prioritizations of the design goals or objectives: $\Sigma\, \omega_r = 1$. This formulation represents a translation of the customer's ranked requirements and goals, via the engineering theories and models underlying the design concept, into a mathematical statement of the design problem.

Complex design problems are almost always solved using some form of decomposition. Decomposition can be brought into the problem formulation by breaking up the vector of design variables into subvectors $\mathbf{x}_i$. If we arrange the elements of the the vector $\mathbf{X}$ in an appropriate sequence of these $\mathbf{x}_i$'s, we obtain a decomposition of the vector $\mathbf{X}$ into the vector $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$.

A design decision process is defined in this context as a set of design decisions

$$\{D_1, D_2, \ldots, D_N\},$$

where each decision D_s can be viewed as a subproblem determining a subvector of the design variables $\mathbf{x}_s$:

Subproblem D_s:

minimize: $\Sigma\, \omega_r\, f_r(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_s, \ldots, \mathbf{x}_N)$

Subject to: $\mathbf{g}_s(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_s, \ldots, \mathbf{x}_N) \leq \mathbf{0}$

$\mathbf{h}_s(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_s, \ldots, \mathbf{x}_N) = \mathbf{0}$,

$\mathbf{x}_s$ varies in the solution of this subproblem. However, all of the other subvectors $\mathbf{x}_t$, $t \neq s$, are fixed, either at an initial value (baseline design), or at a value determined through solution of a decision element sequenced before D_s. The constraint vectors $\mathbf{g}_s$ and $\mathbf{h}_s$ are formed by deleting constraints in which $\mathbf{x}_s$ does not appear from the original constraint vectors $\mathbf{g}$ and $\mathbf{h}$. The subproblem will be an optimization problem if $\mathbf{x}_s$ is explicit in one or more of the multiple objectives $\mathbf{f}_r$. If not, the subproblem reduces to a problem of finding a feasible solution to the equality and inequality constraints (a feasibility problem). These subproblems are identified with design decisions. We approach the study of the design decision-making process by analyzing these decompositions.

Two choices are involved in synthesizing a design methodology to solve the Pareto-optimization problem:

- How to group the design variables into design decisions: which of the set of design variables will constitute each $\mathbf{x}_i$? (choice of a decomposition)
- How to sequence (order) these design decisions--what ordering will we place on the $\mathbf{x}_i$'s in the vector $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$? Which subset will be

addressed first, second, and so on? Which decisions can be made concurrently?

A design decision process must meet various criteria. The most important criterion is that the sequence of design decisions must produce a design that balances the design goals and meets the design requirements (constraints) with a minimum of iteration.

Formulating the problem in terms of optimization is important because it provides a theoretical framework for studying alternative design decision processes. In particular, the analytical notions of convergence and rate of convergence can be introduced to provide quantitative means of evaluating the feasibility and efficiency of a design decision process.

Note that formulation of the problem in terms of optimization theory does not imply that the problem must be solved by numerical optimization methods. The approach of this paper utilizes results from optimization theory to aid in sequencing a set of smaller problems to solve a large design problem. While these subproblems are formulated as optimization or feasibility problems, the details of how these smaller problems are to be solved are not important in this approach. This is a consequence of the fact that the approach given here is based on the Karush-Kuhn-Tucker (KKT) conditions, which depend only on the problem formulation. The KKT conditions are logically completely independent from the technique used to solve the problem. Thus, methods other than numerical (or even graphical) optimization, including engineering judgment, could be applied to yield solutions to these problems, as long as the solutions can be interpreted as being optimal, at least for the Pareto balancing problem. In fact, for life cycle engineering problems, other methods are likely to be necessary, due to need to deal with uncertainty and judgmental factors when considering many downstream design characteristics. The approach presented here is applicable to any design problem that can be posed as a balanced design problem (that is, a Pareto-optimization problem).

Meta-design consists of two components: synthesis and analysis. We address both of these components. We will first discuss analysis, and then show how the analysis tools presented here can be used to aid in synthesis of better design methods.

B. OPTIMIZATION THEORY FRAMEWORK

The issues to be addressed in analyzing the output of a meta-design process (a design decision process) are whether the process converges to a feasible, balanced solution of the original problem, and if so, how much iteration is required to arrive at this solution (how efficient is the process?). Optimization theory provides several concepts that allow these questions to be addressed, which include the following:

- Conditions for maintaining feasibility
- Necessary conditions for a design to be optimal, and
- A technique for quantifying the effect of changing problem parameters on the optimal value for a design goal, subject to feasibility. Problem parameters are design quantities that are determined by considerations outside the scope of a particular design problem.

In the context of a design decision process, problem parameters consist of design variables that are treated as fixed in a particular subproblem of a design decomposition. These variables appear in other subproblems that have been addressed prior to the subproblem currently being addressed.

The conditions for maintaining feasibility are straightforward: the constraints $\mathbf{g(X)}$ must not take on positive values, and the $\mathbf{h(X)}$ must remain zero. The necessary conditions for optimality, developed by Karush, and independently by Kuhn and Tucker, are somewhat more complex. We merely state these conditions here and then use them to develop optimal sensitivity derivatives, a tool for quantifying the effect of changes in problem parameters on the optimal value of a design goal, subject to the constraints.

The KKT conditions are necessary conditions for a particular value $\mathbf{X}^*$ for the vector of design variables $\mathbf{X}$, to be a constrained local minimum. These conditions are

- (Feasibility)

$$\mathbf{g(X) \leq 0}$$

$$\mathbf{h(X) = 0}$$

- (Active constraints)

$$\lambda_j \, g_j(\mathbf{X}) = 0 \quad j = 1,\ldots,m$$

$$\lambda_j \geq 0$$

- (Extremum of the Lagrangian over the primal subspace)

$$\partial F(\omega,\mathbf{X})/\partial x_i + \Sigma\lambda_j \partial g_j(\mathbf{X})/\partial x_i + \Sigma\mu_k \partial h_k(\mathbf{X})/\partial x_i = 0 \qquad i = 1,\ldots,n$$

where m is the number of inequality constraints, n is the number of design variables, and

$$F(\omega,\mathbf{X}) = \Sigma \, \omega_p \, f_p(\mathbf{X}).$$

Using these conditions, we can develop an efficient method for computing the optimal sensitivity derivatives [Ref. 38], which we now define.

The design decisions are related to one another in the following way. Suppose there is a constraint

$$g_j(\mathbf{x}_1,\mathbf{x}_2, \ldots) \leq 0$$

This constraint appears in both design decision D_1 and design decision D_2. If D_1 is to be made before D_2, we must have an initial (baseline) estimate for the values of the variables $\mathbf{x}_2$ before we can evaluate $g_j(X)$. Note that this estimate need not correspond to a feasible design. The values of the variables $\mathbf{x}_1$, as determined by solving D_1, affect the solution for D_2. We say that the design variables $\mathbf{x}_1$ are parameters for D_2, and we distinguish between parameters and local design variables. The $\mathbf{x}_2$'s are local design variables for D_2. The distinction between parameters and local design variables depends on the context: the $\mathbf{x}_1$'s are local design variables for problem D_1.

The optimal solution to D_2 is found by varying the local design variables $\mathbf{x}_2$. The idea of the optimal sensitivity derivative is to evaluate the effect of changes in the parameters $\mathbf{x}_1$ on the optimal value of the objective function that can be achieved through optimization within the design decision element D_2. Let $F(\mathbf{x}_1,\mathbf{x}_2)$, the objective function for D_2, also depend on a vector of parameters $\mathbf{x}_1$. Constraints $\mathbf{g}_2(\mathbf{x}_1,\mathbf{x}_2)$ and $\mathbf{h}_2(\mathbf{x}_1,\mathbf{x}_2)$ may also depend both on the local design variables $\mathbf{x}_2$ and on the parameters $\mathbf{x}_1$. We hold the parameters $\mathbf{x}_1$ fixed while we solve D_2. The optimal solution of D_2 with $\mathbf{x}_1$ fixed defines a function $F^*(\mathbf{x}_1)$, the optimal value function. For a fixed $\mathbf{x}_1$, the value of F^* is given by the minimum of F as $\mathbf{x}_2$ is varied, subject to the constraints of D_2. Since values for the local design variables $\mathbf{x}_2$ are determined in the solution of D_2, F^* is a function of $\mathbf{x}_1$ alone.

We want to compute the derivative $\partial F/\partial x_{1_i}$ subject to certain constraints placed on this derivative, namely

- the constraints of the optimization subproblem D_2 remain satisfied as $\mathbf{x}_1$ is varied, that is

$$\mathbf{g}_2(X) \leq 0$$

$$\mathbf{h}_2(X) = 0$$

- the solution remains optimal as the vector of parameters $\mathbf{x}_1$ is varied.

We can enforce the second restriction by requiring that the KKT conditions remain satisfied. These two restrictions on the derivative may require adjustments in the optimal values of the design variables $\mathbf{x}_2$ to compensate for the changes in the parameters $\mathbf{x}_1$.

It is difficult to strike a compromise between simplicity and precision of notation for the optimal sensitivity derivative. One approach, used frequently in the literature, is based on a notational distinction between the design variables $\mathbf{x}$ and the parameters, p. Then dF/dp represents the optimal sensitivity derivative. We will not need to use the ordinary derivative in the sequel, so one may hope that this compromise will not lead to confusion.

We now compute the optimal sensitivity derivative

$$dF/dp = \partial F/\partial p + \Sigma\, [\partial F/\partial x_i][\partial x_i/\partial p]$$

The function $\mathbf{x}(p)$ describes the changes in the design variables that are required to compensate for the variation of the parameter p. We could approximate the derivatives $\partial x_i/\partial p$ by finite differences, solving the optimization problem for p, determining the optimal solution $\mathbf{x}^*(p)$, and then again for $p + \Delta p$, computing the optimal solution $\mathbf{x}^*(p + \Delta p)$, and then approximating

$$\partial \mathbf{x}/\partial p \sim [\mathbf{x}^*(p + \Delta p) - \mathbf{x}^*(p)]/\Delta p.$$

However, dF/dp can be computed exactly, without solving the optimization problem a second time. We start by applying the requirement that optimality is to be maintained, so we must also satisfy the third KKT condition,

$$\partial F/\partial x_i + \Sigma\lambda_j \partial g_j/\partial x_i = 0$$

Then

$$dF/dp = \partial F/\partial p - \Sigma\, \{\Sigma\lambda_j \partial g_j/\partial x_i\}[dx_i/dp]$$

or, rearranging summations,

$$dF/dp = \partial F/\partial p - \Sigma\, \lambda_j\{\Sigma \partial g_j/\partial x_i[dx_i/dp]\}$$

Now if the active constraint set does not change, and feasibility must be maintained as p is varied, we must ha e, for each j = 1, . . ., m

$$dg_j/dp = \partial g_j/\partial p + \Sigma\, [\partial g_j/\partial x_i][dx_i/dp] = 0.$$

Substituting,

$$dF/dp = \partial F/\partial p + \Sigma\, \lambda_j \partial g_j/\partial p.$$

This formula is of central importance for our study of the design process.

C. GUIDELINES FOR META-DESIGN SYNTHESIS

The results of the preceding section lead to an approach to developing optimal design decision sequences that is outlined in the following paragraphs.

1. Feasible and Optimal Decision Sequences

A design decision D_1 can be made before another design decision D_2 if the values chosen for the design attributes in D_1 do not make D_2 infeasible. For example, suppose that x_1 is a design variable to be determined by decision D_1, and x_2 is also a design variable, to be determined in decision D_2. Suppose x_1 and x_2 are coupled by an inequality constraint g :

$$g(x_1, x_2, \ldots) \leq 0 .$$

In sequencing D_1 and D_2 we have three alternatives:

- Make decision D_1 before D_2. x_1 will then be fixed by D_1 and will be a parameter for decision D_2.
- Make decision D_2 before D_1. x_2 will be a parameter in D_1.
- Combine D_1 and D_2 into a single decision element.

Consider now the case where D_1 is sequenced before D_2. Solution of D_1 will result in a change Δx_1 from the initial value for x_1. The effect of this change on the inequality constraint g can be assessed with a first-order approximation:

$$\Delta g \sim (\partial g/\partial x_1)\, \Delta x_1 .$$

Thus if $(\partial g/\partial x_1)$ and Δx_1 are opposite in sign, Δg will be negative and g will be less critical in making decision D_2 (in comparison with the initial design). If $(\partial g/\partial x_1)$ and Δx_1 have the same sign, g will become more critical for D_2 if we make decision D_1 first.

Feasible sequences for the design decisions can be determined using the directions of proposed changes in the design variables in each decision and the signs of the partial derivatives of inequality constraints coupling two or more decisions together. The criteria are:

F-1) If D_1 does not make (any of) the constraints of D_2 more critical, then D_1 can be sequenced before D_2.

F-2) If D_2 does not make (any of) the constraints of D_1 more critical, then D_2 can be sequenced before D_1.

If D_1 makes the constraints of D_2 more critical, and D_2 makes the constraints of D_1 more critical, combining D_1 and D_2 into a single decision element may be necessary:

If both F-1 and F-2 are met, D_1 and D_2 can be made concurrently.

Many possible decision sequences may meet these criteria. In an extremely tightly coupled problem, all of the initial design decisions may be combined into a single design decision by this procedure. All of the decision sequences meeting criteria F-1 and F-2 will lead to feasible designs. We will next consider additional restrictions on the possible decision sequences, leading to an optimal, and subsequently, a Pareto-optimal or balanced design.

Determination of a sequence of design decisions leading to an optimal design requires an initial suboptimization pass through each of the decision elements. In this suboptimization pass, each of the decisions in which one of the objective functions for the design appears explicitly as a function of the local decision variables is solved in isolation. Parameters for each subproblem, which are in fact local design variables for some other subproblem, are fixed at initial baseline values. The results of the suboptimization pass are then analyzed using sensitivity of optimal solutions to problem parameters. That analysis is used to establish whether an iteration of the decision-making procedure will progress toward an optimal design.

In constructing a decision sequence leading to an optimal design, we again have the three alternatives: place D_1 before D_2 in the decision-making sequence, place D_2 before D_1, or combine them. Let $f(x_1,x_2,\ldots)$ be an objective function to be minimized in both D_1 and D_2. If D_1 is made before D_2, then x_1 appears in D_2 as a parameter. The sensitivity of the optimal solution to D_2 to the parameter x_1 is df/dx_1. We know the directions of proposed changes in the design variables (from the suboptimization pass), so we can determine

$$\Delta f \sim (df/dx_1)\,\Delta x_1.$$

Thus, if df/dx_1 and Δx_1 are opposite in sign, Δf will be negative. Then if D_1 is made before D_2, f will not increase during the decision subsequence $\{D_1,D_2\}$. Any decision subsequence in which f will not increase can form part of an optimizing decision sequence. Optimizing decision sequences are built up from such subsequences, with one additional criterion: decision elements with $df/dx_i = 0$ must be placed after decision elements with $df/dx_i \neq 0$. The need for this criterion emerges from consideration of convergence questions, discussed in Appendix B.

2. Application of Convergence Guidelines to Synthesis of a Design Methodology

Several alternative design methodologies may be available to solve a given problem. Other considerations often enter into the decision sequencing problem, such as controlling costs associated with developing design definition or running product development tests. Thus, in applying the convergence results to synthesize a design methodology, attempting to provide a completely deterministic algorithm for selecting a decision sequence does not make sense. Instead, the following step-by-step process for constructing a design methodology clearly indicates the points at which the design team can select among alternative design methodologies to meet economic, program milestone, product definition technology, or test schedule constraints. The role of optimization theory is to provide well-defined criteria that must be met by these alternative sequences and groupings of design choices.

Step 0. Initialization. Choose an initial design within the variable bounds and make an initial choice of decision elements.

Step 1. Evaluate each decision element to determine an optimal solution for that decision element (in isolation).

Step 2. Identify possible feasible decision sequences. If feasibility requires combination of decision elements, iterate with Step 1.

Step 3. Identify possible optimal decision sequences. Check convergence. If solution is converged, stop. If optimality requires combination of decision elements, iterate with Steps 1 and 2.

Convergence criterion: Both (i) design variables did not change during last solution pass and (ii) all optimal sensitivities are zero ($df/dx_i = 0$ for all parameters x_i) must be satisfied.

Step 4. Select a decision-making sequence that is both feasible and optimal. If D_i is sequenced before D_j, the number of parameters passed from D_i to D_j must equal or exceed the number of independent active constraints common to both decision elements.

Step 5. Find an optimal solution for each decision element in sequence. Update the values of all design variables and iterate from Step 2.

This procedure will converge to an optimal solution from any initial design within the variable bounds, provided that the decision-support procedures applied to solve the individual decision elements do so. The solution set for the procedure is defined by the condition that all $df/dx_i = 0$. In Appendix B, we show that this solution set is the set of KKT points.

In addition, this approach provides a highly efficient technique for finding all of the Pareto-optimal design solutions. Pareto-optimal solutions minimize an objective

$$F = \Sigma\, \omega_r f_r\,, \quad \Sigma\, \omega_r = 1.$$

that is a weighted sum of multiple objectives that may correspond to conflicting requirements. To find all of the Pareto-optimal solutions, one would ordinarily have to solve an optimization problem for each set of values for the weights ω_r.

These steps are not necessary if we use the information developed in the meta-design process. To do so, we allocate the multiple objectives, f_r, to different decision elements. Then, at Step 3, above, we have available the optimal sensitivity derivatives df_r/dp. We define an approximation to the Pareto-optimization problem having optimality conditions

$$dF/dp\,(\omega) = 0.$$

These conditions are identical to the convergence criteria for the solution of the exact Pareto-optimization problem using the meta-design solution procedure. We thus solve the exact Pareto-optimization problem when we satisfy these conditions. In Appendix B, we prove that these conditions may be satisfied by varying the relative prioritizations.

D. APPLICATION TO WATER TANK DESIGN PROBLEM

In this section, we will illustrate the approach presented in the preceding section using the water storage system example presented in Chapter III. In particular, we consider three additional design methodologies for this problem. Our goal in this example is to use a simple problem to illustrate the basic ideas. From one point of view the water storage tank example is too simple: the details of the application of the guidelines outlined in section C above are trivial for each of the design methodologies considered in this section. A slightly more complex example is considered in Appendix C. The development of more comprehensive example applications of the guidelines for design methodology synthesis is a topic for further research efforts.

1. Design Methodologies Based on Optimization

Let c_1 denote capacity; c_2, relative materials cost; l, length dimension of the water storage tank; w, width; and h, height. Consider the following methodologies for solving the water storage tank design problem.

Methodology C

Solve the following optimization problem:

minimize: $(c_1/10 - 1)^2$

subject to:

$$c_1 = lwh$$
$$2(lw + wh + lh) \leq c_2$$
$$c_2 = 6$$
$$c_2, l, w, h \geq \varepsilon > 0, \quad h \leq 2.$$

The design vector decomposition $\mathbf{x}_1 = (c_2)$, $\mathbf{x}_2 = (l,w,h)$ can be used for this problem. We then have decision elements

C_1:

satisfy: $c_2 = 6$

$$2(lw + wh + lh) \leq c_2$$

design variables: c_2

fixed parameters: l,w,h

and

C_2:

minimize:

$$(c_1/10 - 1)^2$$

subject to:

$$c_1 = lwh$$
$$2(lw + wh + lh) \leq c_2$$
$$l, w, h \geq \varepsilon > 0, \quad h \leq 2$$

design variables: c_1,l,w,h

fixed parameters: c_2

Since there is a unique objective function for the problem addressed by Methodology C, $(c_1/10-1)^2$, we need not define weighting factors.

Methodology D

Solve the following optimization problem:

minimize: $(c_2/6 - 1)^2$

subject to:

$$c_2 = 2(lw + wh + lh)$$
$$lwh \geq c_1$$
$$c_1 = 10$$
$$c_1, l, w, h \geq \varepsilon > 0, \quad h \leq 2.$$

The design vector decomposition $\mathbf{x}_1 = (c_1)$, $\mathbf{x}_2 = (l,w,h)$ can be used for this problem. We then have decision elements

D_1:

satisfy: $c_1 = 10$

$lwh \geq c_1$

design variables: c_1

fixed parameters: l,w,h

and

D_2:

minimize:

$$(c_2/6 - 1)^2$$

subject to:

$$c_2 = 2(lw + wh + lh)$$

$$lwh \geq c_1$$

$$l,w,h \geq \varepsilon > 0, \quad h \leq 2$$

design variables: c_2, l,w,h

fixed parameters: c_1

Again, there is a unique objective function for the optimization problem addressed by Methodology D, $(c_1/6\text{-}1)^2$, so it is not necessary to define weighting factors.

Methodology E

Solve the following optimization problem:

minimize:

$$\omega_1 [c_1/10 - 1]^2 + \omega_2 [c_2/6 - 1]^2$$

subject to:

$$lwh \geq c_1$$

$$2(lw + lh + wh) \leq c_2$$

$$\omega_1 + \omega_2 = 1$$

$$c_1, c_2, l,w,h \geq \varepsilon > 0, \quad h \leq 2$$

$$\omega_1,\omega_2 \geq 0$$

The design vector decomposition $\mathbf{x}_1 = (\omega_1,\omega_2)$, $\mathbf{x}_2 = (c_1,c_2,l,w,h)$ can be used for this problem. We then have decision elements

E_1:

minimize:

$$\omega_1 [c_1/10 - 1]^2 + \omega_2 [c_2/6 - 1]^2$$

subject to:

$$\omega_1 + \omega_2 = 1$$

$$\omega_1,\omega_2 \geq 0$$

$$c_1,c_2 \text{ fixed parameters}$$

and

E_2:

minimize:

$$\omega_1 [c_1/10 - 1]^2 + \omega_2 [c_2/6 - 1]^2$$

subject to:

$$lwh \geq c_1$$

$$2(lw + lh + wh) \leq c_2$$

$$c_1, c_2, l, w, h \geq \varepsilon > 0, \ h \leq 2$$

$$\omega_1, \omega_2 \text{ fixed parameters}$$

If we interpret some of the requirements identified in the water tank design problem as goals, we can model them in the context of optimization theory as objective functions. Thus, the objective functions in methodologies C, D, and E are stated as minimization of a the deviation of a product characteristic from a desired target or goal value. Other formulations are certainly possible. For example, we might state the objective for methodology E as

minimize:

$-\omega_1 c_1/10 + \omega_2 c_2/6$,

minimizing cost and maximizing capacity. In the case where both requirements cannot be met simultaneously, both the direct minimization and goal formulations give similar results. Note that the direct minimization formulation is preferable if it is not known that the requirements are incompatible.

Requirements that cannot be relaxed are modelled in optimization theory as constraints. Taking advantage of this, we apply optimization theory, specifically convergence theory, to assess the capability of the remaining design methodologies to solve the problem. The details of such an approach are given in Appendix B, and applied to develop a design methodology for landing gear layout in Appendix C. The basic concepts are illustrated in this chapter. Again, we emphasize that this application of optimization theory to assess a design methodology is distinct from the application of design optimization methods, or more specifically numerical optimization, as a part of a particular design methodology.

Optimization theory is applied to evaluate the suitability of a design methodology to meet a set of requirements by considering each step in the design methodology to be a decision element. In the context of optimization theory, decision elements are optimization or feasibility subproblems. These subproblems are related to one another by engineering theories and models linking design attributes in distinct decision elements. In evaluating the suitability of a design methodology to meet requirements, engineering theories are analyzed to determine the monotonicity of the relationships among attributes implied by the engineering theories and models. Thus, for example, if width is increased with length and height fixed, an increase in capacity is required to satisfy the engineering theory

$$c_1 = lwh.$$

Thus, capacity is monotonically increasing with width when the "capacity" engineering theory is enforced.

In a design methodology with multiple steps, the values of design attributes will be changed as decisions are made. The monotonicity information can be used to determine whether these decisions will adversely affect the feasibility of subsequent decisions. The monotonicity information can also be used to assess the overall progress of the decision-making sequence toward an optimal, or alternatively, toward a Pareto-optimal (balanced) design. Convergence of a methodology to a feasible design and progress toward a balanced or optimal design can be ensured by imposing certain criteria on the sequence of decision elements. The simplest approach is to allow decision element D_i to be sequenced before decision element D_j only if the choices for values of design attributes in decision element D_i will not adversely affect feasibility or optimality of decision element D_j. Convergence of such an approach is considered from a theoretical point of view in Appendix B. Of course, such a sequence may not be possible to realize in practice. An alternative approach is then to constrain prior decision elements so that subsequent decision elements have feasible solutions.

To illustrate these ideas, consider methodology B of Chapter III. In methodology B, choices for height, capacity, and cost are distinct decision elements that are sequenced before a choice is made for the value of the width design attribute. Choice of the width attribute is in fact constrained by $w \geq 0$. Clearly, it is possible to choose values for the height, cost, and capacity attributes that make the width decision element infeasible. (for example, height = 2, cost = 6 and capacity = 10). Thus, using the concept that prior decisions should not adversely affect feasibility of subsequent decisions, we are able to accurately identify one of the limitations of methodology B. Methodology B has additional limitations. One of these limitations is the fact that methodology B does not provide any means to continually improve the design in terms of cost and capacity goals throughout the design process. To evaluate methodologies attempting to accomplish such improvements, we must consider optimality, or equivalently, Pareto-optimal balance in addition to feasibility.

Methodologies C and D represent different approaches to the design optimization problem. In methodology C, if we attempt to solve C_2 before C_1, we may not be able to satisfy the constraint $c_2 = 6$. The sequence $\{C_1, C_2\}$ is feasible: once the cost is fixed in C_1, the optimization problem C_2 has a feasible solution for any value of cost $\geq 6\varepsilon^2$. Since the choice of ε is arbitrary, the sequence $\{C_1, C_2\}$ is feasible for any positive value of cost. The analogy that methodology D is feasible if the capacity is positive is valid.

Do methodologies C and D lead to optimal or balanced designs? The answer to this question depends on who defines optimality. Optimality must be defined by the customer's needs. Thus, methodologies C and D can be optimizing only when they match the customer's ranking of the cost and capacity requirements. Methodology C provides the maximum capacity meeting the cost requirement, and methodology D delivers the minimum cost to meet the capacity requirement. This fact is reflected in the sequence of design decisions. In fact, the customer may need to balance cost and capacity in some sense. Neither methodology C or methodology D can address this balancing problem. Thus, we have to reject methodologies C and D if we wish to produce designs balancing cost and capacity.

Considerable additional complexity is required to fully address this problem of balanced design, as is illustrated by methodology E. Methodology E calls for a separate requirements ranking decision element (decision element E_1) in which relative weights for the cost and capacity goals are determined. Incorporation of this decision element ensures that methodology E will deliver balanced designs. In the second decision element in methodology E, (decision element E_2), we determine values of cost and capacity goals that result in a feasible solution of the Pareto-optimization problem.

The optimization problem in methodology E is stated so that we can always find a solution: even though we may not meet the cost or capacity goals, the design will balance the degree to which those goals are achieved, with relative priorities determined by the weighting factors. Thus, the statement of methodology E ensures feasibility in this restricted sense. We conclude that methodology E is well-matched to the water tank design problem. Unfortunately, we have paid too high a price for this suitability: we must ask the customer to prioritize cost and capacity (i.e., choose values for the weighting factors, ω_i) without the benefit of information about the design relationship between achievable values of cost and capacity--a question of comparable difficulty to the design problem itself.

Perhaps a more practical approach is to generate the cost/capacity curve and relate this curve to the relative prioritizations. This information can be used in a requirements negotiation. Information to support requirements negotiation is most valuable before we completely define the design. Once we reach agreement on the goals and their priorities, we can then address the design of the water storage tank itself. Methodology E is not well suited to this expanded problem. To generate the cost/capacity curve using methodology E, we would have to vary the relative prioritizations, and execute a reoptimization of decision element E-2 for each set of priorities. Thus, we would have to design many water storage tanks before we could negotiate with the customer on what goals for the water storage tank should be. In more complex design problems, each reoptimization, in itself, may involve the execution of a complete design methodology in a decision element such as E_2.

2. A Design Methodology to Support Requirements Negotiation

An approach for efficiently generating families of Pareto-optimal solutions can be developed by further extending the application of optimization theory to design methodologies consisting of separate decision elements. In an optimization-based theory of design methodologies, we can pose the following question: how can we generate information to support requirements negotiation by executing relatively simple design methodologies, comparable to methodology C or D? This question is answered by a method based on the meta-design technique. Optimal sensitivity derivatives can be used to avoid having to re-execute the simple design methodology for each combination of values of attributes that may be of interest in the requirements negotiation process.

These insights begin with the observation that optimization theory can be applied to determine convergence of a sequential decision-making process. Looking at the problem in this light, we establish the convergence of a parameter passing scheme that allows us to separate the multiple objective functions, locating them in distinct subproblems. Convergence is ensured through the use of optimal sensitivity derivatives. Finally, we show how optimality conditions for the full Pareto-optimization problem are related to the optimal sensitivity derivatives of the individual objective functions of the subproblems. The details of this argument are developed in Appendix B.

We illustrate the concepts behind the method by giving an improved methodology for the water tank design problem. Although this problem can easily be solved explicitly, we solve the problem using Lagrange multipliers and optimal sensitivity derivatives, techniques which can be readily applied to more complex problems in life cycle engineering. A somewhat more complex example, balancing development risk and performance in an aircraft sizing problem, is worked out in detail in Appendix D.

We consider the following methodology for developing design information to support requirements negotiation:

Methodology F

Solve the following optimization problem (the same problem as solved by methodology E):

minimize:

$$\omega_1 [c_1/ 10 - 1]^2 + \omega_2 [c_2/6 - 1]^2$$

subject to:

$$lwh \geq c_1$$

$$2(lw + lh + wh) \leq c_2$$

$$\omega_1 + \omega_2 = 1$$

$$c_1, c_2, l,w,h \geq \varepsilon > 0, \; h \leq 2$$

$$\omega_1,\omega_2 \geq 0$$

The design vector decomposition $\mathbf{x}_1 = (c_1)$, $\mathbf{x}_2 = (c_2)$, $\mathbf{x}_3 = (\omega_1,\omega_2,l,w,h)$ can be used for this problem. We formulate decision elements as follows.

F_1:

minimize:

$$f_1 = (c_1/10 - 1)^2$$

subject to:

$$c_1 - lwh \leq 0$$

design variables: c_1

l,w,h fixed parameters

and

F_2:

minimize:

$$f_2 = (c_2/6 - 1)^2$$

subject to:

$$2(lw + wh + lh) - c_2 \leq 0$$

design variable: c_2

l,w,h fixed parameters

F_3:

minimize:

$$F = \omega_1 f_1^{opt}(l,w,h) + \omega_2 f_2^{opt}(l,w,h)$$

subject to:

$$h - 2 \leq 0$$

$$\omega_1 + \omega_2 = 1$$

$$\omega_1, \omega_2 \geq 0$$

where the design variables are now: ω_1,ω_2,l,w,h. f_1^{opt} and f_2^{opt} are the optimal values of the objective functions for subproblems F_1 and F_2, respectively, with l,w,and h fixed.

Although methodology F is similar in some ways to methodology E, there is an important difference in that we have split up the multiple objective functions and assigned them to different subproblems. Although the sequences of design decisions which make sense for methodology F are somewhat restricted (F_1 and F_2 must be executed before F_3 in order to define f_1^{opt} and f_2^{opt}), methodology F provides us with a very efficient technique for constructing the cost-capacity curve as a function of the requirements priorities. We now work through the solution of the requirements negotiation problem using methodology F.

The basic concept is that we can obtain the optimality conditions for subproblem F_3 directly from the solutions to subproblems F_1 and F_2. Thus, we do not actually have to solve F_3. As is shown in Appendix B, the formulation of F_3 is such that the optimality conditions for F_3 are the same as the optimality conditions for the undecomposed optimization problem addressed by both methodology E and methodology F. Thus, the particular decomposition strategy used in methodology F allows us to solve the Pareto-optimization problem indirectly, using the solutions to two suboptimization problems (subproblems F_1 and F_2). Although the difference between this solution technique and methodology E is inconsequential for the simple water storage tank design problem, methodology F can be applied to much more complex design problems with a few relatively simple extensions. It is rarely, if ever, practical to apply the approach of methodology E to complex design problems.

Solution of Requirements Negotiation Problem using Methodology F.

The formulation of subproblem F_3 in terms of f_1^{opt} and f_2^{opt} suggests the use of optimal sensitivity derivatives. Thus, in applying methodology F to the requirements negotiation problem, the first step is to solve subproblems F_1 and F_2, estimating the optimal sensitivity derivatives of f_1^{opt} and f_2^{opt} with respect to l, w, and h from the solutions to these subproblems. To do this, we need to find the values for the Lagrange multipliers for the constraints in which l, w, and h appear explicitly.

Solving F_1 to determine a value for the Lagrange multiplier of the constraint

$$g_1 = c_1 - lwh,$$

we note that an optimal solution to this problem must satisfy the third KKT condition:

$$\partial f_1/\partial c_1 + \lambda_1 \partial g/\partial c_1 = 2(c_1/10 - 1)(1/10) + \lambda_1 = 0.$$

Then

$$\lambda_1 = (1/5)(1 - c_1/10).$$

where λ_1 is the desired Lagrange multiplier.

Determining a value for the Lagrange multiplier of the constraint

$$g_2 = 2(lw + wh + lh) - c_2$$

appearing in subproblem F_2, we again apply the third KKT condition

$$\partial f_2/\partial c_2 + \lambda_2 \partial g/\partial c_2 = (1/3)(c_2/6 - 1) - \lambda_2 = 0.$$

Thus,

$$\lambda_2 = (1/3)(c_2/6 - 1).$$

We now have the information we need to solve subproblem F_3. The optimal solution to problem 2(a) is a function of l, w, and h. Denote this function by $f_1^{opt}(l,w,h)$. Compute the optimal sensitivity derivatives $D_l f_1^{opt}$, $D_w f_1^{opt}$, and $D_h f_1^{opt}$. (The optimal sensitivity derivatives can be computed from partial derivatives of the objective function and constraints with respect to the decision variables at the optimal solution.) To emphasize that l, w, and h are parameters for subproblems F1 and F2, denote them by

$$l = p_1,\ w = p_2, \text{ and } h = p_3.$$

Use the optimal sensitivity derivatives to construct an approximation

$$f_1^{opt}(p_1,p_2,p_3) \sim f_1^{opt}(p_1^0,p_2^0,p_3^0) + \Sigma\, D_{p_i} f_1^{opt} \Delta p_i$$

about the point (p_1^0,p_2^0,p_3^0).

Define $f_2^{opt}(p_1,p_2,p_3)$ in the same way, and approximate

$$f_2^{opt}(p_1,p_2,p_3) \sim f_2^{opt}(p_1^0,p_2^0,p_3^0) + \Sigma\, D_{p_i} f_2^{opt} \Delta p_i\,.$$

Applying the technique of optimal sensitivity derivatives to differentiate the optimal value of f_1 as the parameter p_1 is varied,

$$D_{p_1} f_1^{opt} = \partial f_1/\partial p_1 + \Sigma_j\, \lambda_j\, \partial g_j/\partial p_1 = \lambda(-wh) = (1/5)(1 - c_1/10)(-wh)$$

where λ is the Lagrange multiplier of the constraint $g = c_1 - lwh \le 0$ in problem 2(a).

In a similar computation, determine

$$D_{p_2} f_1^{opt} = (1/5)(1 - c_1/10)(-lh)$$

and

$$D_{p_3} f_1^{opt} = (1/5)(1 - c_1/10)(-lw)$$

Then

$$f_1^{opt}(p_1,p_2,p_3)$$

$$\sim f_1^{opt}(p_1^0,p_2^0,p_3^0) + (1/5)(1 - c_1/10)(-wh)\Delta p_1 + (1/5)(1 - c_1/10)(-lh)\Delta p_2$$

$$+ (1/5)(1 - c_1/10)(-lw)\Delta p_3.$$

Similar computations give

$$D_{p_1} f_2^{opt} = (1/3)(c_2/6 - 1)(w+h)$$

(the other derivatives are computed in exactly the same way), and

$$f_2^{opt}(h_0,c_1)$$

$$\sim f_2^{opt}(p_1^0,p_2^0,p_3^0) + (1/3)(c_2/6 - 1)(w+h)\Delta p_1 + (1/3)(c_2/6 - 1)(l+h)\Delta p_2$$

$$+ (1/3)(c_2/6 - 1)(l + w)\Delta p_3$$

Using the approximations to $f_1^{opt}(,p_1,p_2,p_3)$ and $f_2^{opt}(p_1,p_2,p_3)$, we determine capacity = $c_1(\omega_1,\omega_2,p_1,p_2,p_3)$ and cost = $c_2(\omega_1,\omega_2,p_1,p_2,p_3)$ as solutions to the minimization problem posed for subproblem F_3, which we repeat here.

minimize:

$$F = \omega_1 f_1^{opt}(p_1,p_2,p_3) + \omega_2 f_2^{opt}(p_1,p_2,p_3)$$

subject to:

$$p_3 - 2 \leq 0$$

$$\omega_1 + \omega_2 = 1$$

$$\omega_1,\omega_2 \geq 0$$

where the design variables are now:

$$l = p_1,\ w = p_2, \text{ and } h = p_3.$$

Optimality conditions for this problem are

$$\partial F/\partial p_1 = \omega_1 D_{p_1} f_1^{opt} + \omega_2 D_{p_1} f_2^{opt} = 0$$

$$\partial F/\partial p_2 = \omega_1 D_{p_2} f_1^{opt} + \omega_2 D_{p_2} f_2^{opt} = 0$$

$$\partial F/\partial p_3 + \lambda = \omega_1 D_{p_3} f_1^{opt} + \omega_2 D_{p_3} f_2^{opt} + \lambda = 0$$

The key to methodology F is to fix the optimal sensitivity derivatives and regard these equations as determining values for ω_1 and ω_2 that correspond to those values of the optimal sensitivity derivatives, allowing us to bring the theory of Appendix B to bear on the minimization problem. In Appendix B, we show that sequential parameter passing schemes of the type exemplified by methodology F converge to the optimal solution of a Pareto-optimization problem such as the problem approximated in this step (step 4). The benefit accruing from this approach is that we can determine the entire family of Pareto-optimal solutions (corresponding to different values for the ω_i's) using only

- Information about the solution to an optimization problem corresponding to each of the objective functions, and
- Partial derivatives.

The alternative approach (methodology E) would require us to solve a separate optimization problem for each combination of values of the ω_i's we wish to consider.

Continuing with the solution of the optimality conditions for subproblem F_3, we have 5 unknowns:

$\omega_1, \omega_2,$ l, w, and h,

and 4 independent equations (five of which are nonlinear):

$\omega_1 + \omega_2 = 1$

(3 optimality conditions).

Thus, we can characterize the entire family of Pareto-optimal solutions by varying one of the parameters. One way to do this is to solve:

$$\partial F/\partial p_1 = \omega_1 D_{p_1} f_1^{opt} + \omega_2 D_{p_1} f_2^{opt} = 0$$

$$= \omega_1 (1/5)(1 - c_1/10)(-wh) + \omega_2 (1/3)(c_2/6 - 1)(w+h)$$

to obtain

$$\omega_1 = (1/3)(c_2/6 - 1)(w+h)/[(1/3)(c_2/6 - 1)(w+h) + (1/5)(1 - c_1/10)(wh)].$$

We apply the remaining two optimality conditions to determine that l = w and h = min (w,2). To present our results, we return to the engineering theories and models relating cost and capacity to l, w, and h. These determine the achievable values for cost and capacity. The variables c_1 and c_2 of subproblems F_1 and F_2 refer to the cost and capacity constraints. The difference between these constraints and the achievable values is, of course, the whole point of requirements negotiation.

We can vary w, determining cost, capacity and ω_1 as w is varied. The relationship between achievable capacity, actual cost, and requirements prioritization, ω_1 obtained in this way is plotted in Figure IV-1. Since the cost and capacity requirements cannot both be met, the customer must accept some loss of capacity, cost, or both. The magnitude of the loss depends on the relative prioritization given to achieving each of the goals. The loss in capacity is defined as

capacity goal - achieved capacity

and represents how far we are from achieving the desired capacity of 10 ft^3. The loss in cost is defined as

actual cost - cost goal,

adopting the convention that losses are positive.

This information can then be used to negotiate values for the relative prioritizations ω_1 and $\omega_2 = 1 - \omega_1$. Once these values are fixed, the optimal values for the remaining design parameters are also determined.

In most cases, where the problem posed in methodology E cannot be solved explicitly, this procedure will be much more efficient than methodology E. This is a consequence of the fact that the optimization problem in methodology E must be solved a large number of times, one solution for each value of ω_1, to determine the effect of ω_1 and ω_2 on cost/capacity relationship, while methodology F can generate the entire cost capacity relationship from the solutions of a few optimization problems, one for each of the multiple objective functions.

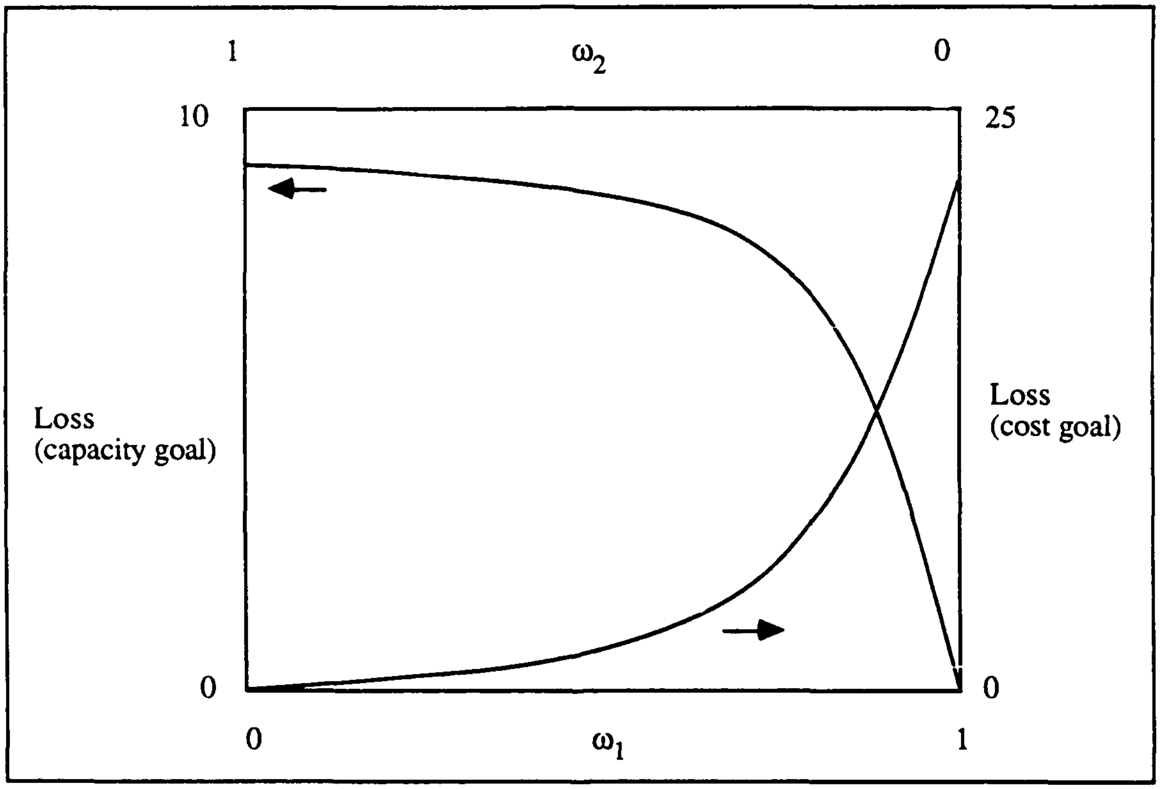


Figure IV-1. "Loss" of Cost and Capacity Goals vs. Requirements Priorities for a Water Storage Tank

3. Verification of the Methodology F Solution using Methodology E.

The water storage tank example is simple enough so that an explicit solution using the straightforward, though inefficient, methodology E can be found. We now illustrate this solution, in part to motivate the results of Appendix B, and in part to illustrate, in detail, the differences between methodologies E and F.

To use methodology E, we must be able to solve the optimization problem:

minimize:

$$\omega_1 [c_1/10 - 1]^2 + \omega_2 [c_2/6 - 1]^2$$

subject to:

$$lwh \geq c_1$$
$$2(lw + lh + wh) \leq c_2$$

design variables:

$$c_1, c_2, l, w, h > 0, \; h \leq 2$$

for fixed values of ω_1 and ω_2, with $\omega_1 + \omega_2 = 1$. The optimality conditions are as follows:

The Lagrangian is:

$$L(\mathbf{x}, \lambda) =$$

$$\omega_1 [c_1/10 - 1]^2 + \omega_2[c_2/6 - 1]^2 + \lambda_1(c_1 - lwh) + \lambda_2(2(lw + lh + wh) - c_2) + \lambda_3(h-2)$$

The design variables must satisfy:

(i) $\partial L/\partial c_1 = (1/5)\omega_1[c_1/10 - 1] + \lambda_1 = 0$

(ii) $\partial L/\partial c_2 = (1/3)\omega_2[c_2/6 - 1] - \lambda_2 = 0$

(iii) $\partial L/\partial l = \lambda_1(-wh) + 2\lambda_2(w + h) = 0$

(iv) $\partial L/\partial w = \lambda_1(-lh) + 2\lambda_2(l + h) = 0$

(v) $\partial L/\partial h = \lambda_1(-lw) + 2\lambda_2(l + w) + \lambda_3 = 0$

We can solve equation (i) for λ_1 and equation (ii) for λ_2:

$$\lambda_1 = (-1/5)\omega_1[c_1/10 - 1]$$
$$\lambda_2 = (1/3)\omega_2[c_2/6 - 1]$$

Substituting these two expressions into equation (iii), we obtain:

$$(-1/5)\omega_1[c_1/10 - 1](-wh) + (1/3)\omega_2[c_2/6 - 1](w + h) = 0$$

Since we must have $\omega_1 + \omega_2 = 1$, we can set $\omega_2 = 1 - \omega_1$ and solve this equation for ω_1:

$$(-1/5)\omega_1[c_1/10 - 1](-wh) + (1/3)(1 - \omega_1)[c_2/6 - 1](w + h) = 0$$

$$-\omega_1\{(1/5)[c_1/10 - 1](-wh) + (1/3)[c_2/6 - 1](w + h)\}$$
$$+ (1/3)[c_2/6 - 1](w + h) = 0$$

$$\omega_1 = (1/3)(c_2/6 - 1)(w+h)/[(1/3)(c_2/6 - 1)(w+h) + (1/5)(1 - c_1/10)(wh)].$$

This is precisely the solution delivered by methodology F.

E. CONCLUSIONS

Analysis of convergence using optimization theory can be used to guide the synthesis of a design methodology delivering product specifications balancing conflicting requirements. The applicability of the KKT conditions depends on the nature of the problem statement and is independent of any technique, numerical or other, for solving the problem. The distinction between the statement of a design problem and the techniques for solving such problems is fundamental to our subject. In design, we can pose many problems as optimization problems. It is not always practical to solve these problems using numerical techniques. However, all complex design problems are solved in practice using some form of decomposition. Since the decomposition technique developed in this paper is based directly on the KKT conditions, application of this method is not dependent on use of numerical techniques for design optimization.

The methods of Appendix B can be used to prove that a sequential design process will converge to an optimal or balanced solution. Thus, this approach can be used to extend the application of optimization ideas to areas of design where numerical optimization techniques have been difficult to apply. Examples of such areas include design problems in which rough approximations must be made in the engineering theories and models. In such cases, engineering judgment, perhaps based on comparison of predictions made by several approximate theories, is required to determine values for design variables.

The approach presented in this chapter also promises to be useful in engineering design problems having attributes that are subject to uncertainty or random variations. In this context, the results offer a way to systematically extend the optimization methods of Taguchi to solve complex design problems encountered in the development of advanced technology systems.

A third area of application of the method is in the development of computational environments for design using object-centered programming and constraint propagation. This idea is developed in Appendix A.

While the approach is not dependent on use of numerical optimization, this is not to say that these ideas are not highly relevant to problems of numerical optimization. The approach taken here is particularly relevant to decomposition techniques for solving a large optimization problem by defining an iteration scheme on a network of smaller optimization problems. The results of Appendix B represent a first step toward a convergence theory for such decomposition methods. It is pertinent to point out that, in significant measure, the recent advances in algorithms for numerical optimization have come about through the application of the global convergence theory and analysis of asymptotic convergence rates. Thus, one can hope to see corresponding improvements in the decomposition methods through further development of the theory along the lines laid out in Appendix B.

The difficulty of determining the complete family of Pareto-optimal solutions to an engineering design problem is well known, as is the value of this information in the requirements negotiation process. The method for determining the entire family of Pareto-optimal solutions from the solution of a finite number of single objective optimization problems and partial derivatives, presented in this chapter, is particularly valuable in that it represents an efficient approach to what is normally a computationally intensive problem.

V. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

The initiatives noted in Chapter I have successfully defined the problem of early consideration of downstream product attributes such as reliability, maintainability, supportability, and producibility in the weapon system design process. From the point of view of the system developer, however, these initiatives have added little or nothing to the methods available to solve complex life cycle engineering problems. Initiatives to integrate CAD/CAE tools, pursued in the absence of a design methodology describing how such tools are to be used after integration, are unlikely to result in significant progress toward realization of ULCE. Research in design methodology is crucial for the accomplishment of the aims of life cycle engineering.

Meta-design, as defined in this paper, is a systematic approach to the development of design methodologies. Two existing approaches to meta-design, ISM and DSS, use a directed graph model of the design problem as input . However, the directed graph does not contain sufficient information to allow development of a methodology capable of resulting in a balanced, feasible design. Such a methodology usually requires as input the full analytical content of the design problem. A similar problem faces the user of QFD, another tool for structuring and tracking decision processes. To be used successfully in complex, advanced technology systems development, QFD must be coupled with a design methodology that takes into account the analytical aspects of the design problem.

This paper presents a technique for using results from optimization theory to aid in accomplishing the meta-design process. This technique facilitates evaluation of existing design methodologies to determine the extent that they will lead to balanced, feasible designs. The technique may also be used to support synthesis of new design methodologies.

In addition, a new technique for developing design information to support negotiation of conflicting requirements has been introduced. This capability is critical in cases where the initial requirements levied on the design team conflict, and information must be developed to allow rational trade-offs. Simple methods, such as ISM, DSS, and QFD do not support this activity.

B. RECOMMENDATIONS FOR FUTURE RESEARCH

This paper underscores the foundational importance of design theory and methodology development for ULCE. Development of ULCE design methodologies is a research area that has received very little attention. Significant progress towards implementation of ULCE is not likely to occur until this crucial area is addressed. The concept of meta-design discussed in this paper is a key element in development of an ULCE environment that represents a flexible design system. Such a system, if implemented, could revolutionize design team productivity and design quality. In moving toward such a goal, additional research should be considered in certain areas.

First, the methods of this paper should be demonstrated and evaluated by applying them to a design problem comparable in scope to the conceptual development of aircraft system. In particular, a life cycle approach to this problem should be formulated and executed through application of this methodology.

Research into the application of the ideas contributed by this paper to product development and system management problems characterized by uncertainty or random variations in the design attributes should be also be pursued. An area of considerable interest is the integration of the approach of this paper with QFD and Taguchi methods to rapidly develop robust designs for complex advanced technology systems. QFD can be used to structure the systems engineering process leading to development of the input information (the system life cycle concept) needed for meta-design. Taguchi methods, as generalized by Tse [Ref. 24], can then be applied as a method for solving the optimization subproblems identified by the meta-design approach.

Finally, the convergence theory developed in this paper for decomposition methods of optimization should receive further investigation. Useful results concerning the convergence of iterative decomposition methods that are not sequential are immediately accessible using the techniques developed here.

REFERENCES

1. *Implementation Plan for Unified Life Cycle Engineering*, Air Force Systems Command, March 1987.
2. ML Brei, et al., *Architecture and Integration Requirements for an ULCE Design Environment*, IDA Paper P-2063, April 1988.
3. "Concurrent Engineering -- A Total Quality Management Process", Memorandum for Secretaries of the Military Departments, Dr. Robert B. Costello, Undersecretary of Defense, Acquisition, 9 March 1989.
4. *Operational Readiness with High Performance Systems*, Report of the Defense Science Board Summer Study Panel, April 1982.
5. *Transition from Development to Production*, DoD Manual 4145.7M, U.S. Department of Defense, Director of Defense Research and Engineering, April 1985.
6. *Best Practices -- How to Avoid Surprises in the World's Most Complicated Technical Process*, The Transition from Development to Production, Department of the Navy, NAVSO P-6071, March 1986.
7. *System Engineering Management Guide*, Defense Systems Management College, December 1986.
8. *USAF R&M 2000 Process*, Air Force Pamphlet AFP 800-7, 1 January 1989.
9. *Better Designs in Half the Time*, Bob King, GOAL/QPC, 1987.
10. *Product Design for Assembly*, G. Boorthroyd and P. Dewhurst, Booythroyd-Dewhurst, Inc., 1987.
11. *Introduction to Quality Engineering: Designing Quality into Products and Processes*, Genichi Taguchi, Unipub-Krause International Publications, 1986.
12. *Report of the Joint Industry-DoD Task Force on Computer Aided Logistics Support (CALS)*, Frederick R. Riddell, et al., Institute for Defense Analyses Report R-285 (5 volumes),June 1985.
13. *The Role of Concurrent Engineering in Weapon Systems Acquisition*, Robert I. Winner, et al., Institute for Defense Analyses Report R-338, December 1988.
14. *Total Quality Management -- a Guide for Implementation*, Office of the Deputy Assistant Secretary of Defense, Total Quality Management (OASD (P&L) TQM), DOD 5000_51-G, Draft, 23 March 1989.
15. *Out of the Crisis*, W. Edwards Deming, MIT Center for Advanced Engineering Study, Cambridge, MA, 1982.
16. *Juran on Planning for Quality*, J.M. Juran, The Free Press, New York, 1988.
17. *Systems Engineering and Analysis*, Benjamim S. Blanchard, and Wolter J. Fabrycky, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1981.

18. *Designing Defense Systems - The Trade-Off Process,* Wilbur V. Arnold, Defense Systems Management College, Technical Management Department, November 1986.

19. "Application of ULCE to Design Quality for Competitive Edge," Siegfried Goldstein, (slides for presentation given at RAMCAD Technical Interchange Meeting Number 3, Problems and Solutions in RAMCAD Development and Application), Institute for Defense Analyses, April 22-23, 1987.

20. "The Skunk Works' Management Style -- It's No Secret", Wright Brothers Lecturship in Aeronautics, Ben R. Rich, Advanced Development Projects, Lockheed Aeronautical Systems Company, American Institute for Aeronautics and Astronautics, AIAA-88-4516, Atlanta, GA, September 7-9, 1988.

21. *Characterizing the Acquisition Process*, Leonard Sullivan, Jr., January 1986.

22. Remarks by Dr. Daniel Whitney at the National Science Foundation (NSF) Design Theory and Methodology Grantee's Workshop, University of Massachusetts, Amherst, MA, June 1989.

23. *Partitioning and Planning for Unified Life Cycle Engineering Design: A Conceptual Exposition*, D. Muster, J.K. Allen, and F. Mistree, DM&A Incorporated, Houston, September 1989.

24. *Management of Risk and Uncertainty in Product Development Processes*, Edison Tse and William Cralley, Institute for Defense Analyses, Paper P-2153, June 1989.

25. "A Flexible Aid for Conceptual Design based on Constraint Propagation and Component-Modelling," M.A. Kolb, AIAA-88-4427, AIAA/AHS/ASEE Aircraft Design, Systems, and Operations Meeting, Atlanta, GA, September 7-9, 1988.

26. "The Application of Artificial Intelligence Technology to Aeronautical System Design," E.E. Bouchard, J.E. Rogan, and G. Kidwell, AIAA-88-4426, AIAA/AHS/ASEE Aircraft Design, Systems, and Operations Meeting, Atlanta, GA, September 7-9, 1988.

27. Sullivan, L.P., *Quality Function Deployment,* Quality Progress, June 1986, pages 39-50.

28. Ross, Philip J., *Taguchi Techniques for Quality Engineering,* McGraw-Hill, NY, NY, 1988.

29. Malone, D.W., *An Introduction to the Application of Interpretive Structural Modeling*, Proc. IEEE, Volume 63, Number 3, March 1975, pages 397-404.

30. Warfield, J.N., ed., *Annotated Bibliography of Publications*, Institute for Advanced Study in the Integrative Sciences, George Mason University, December 1986.

31. Akagi,S., R. Yokoyama, R., and K. Ito, *Optimal Design of Semisubmersible's Form Based on Systems Analysis*, Trans. ASME, Voume 106, December 1984, page 524.

32. Steward, D.V., *The Design Structure System: A Method for Managing the Design of Complex Systems*, IEEE Trans. on Eng. Mgmt., Volume EM 28, Number 3, August 1981.

33. Carré, Bernard, *Graphs and Networks*, Clarendon Press, Oxford, UK, 1979.

34. Lockheed Missiles and Space Company, *System Engineering Management Guide*, Defense Systems Management College, Ft. Belvoir, VA. Contract MDA 903-82-C-0339, October 1983.

35. Padula, S. L., C. Sandridge, R. Haftka, and J.L. Walsh: *Demonstration of Decomposition and Optimization in the Design of Experimental Space Systems*, Second NASA/Air Force Symposium on Recent Advances in Multidisciplinary Analysis and Optimization", Hampton, VA, September 28-30, 1988.

36. Rogers, James L.: *A Knowledge-based Tool for Multilevel Decomposition of a Complex Design Problem* , NASA TP-2903, NASA Langley Research Center, Hampton, VA, May 1989.

37. Cohon, J.L.; *Multiobjective Programming and Planning* , Academic Press, New York, 1978.

38. Sobieszczanski-Sobieski, J., J.F.M. Barthelemy, and K.M. Riley, *Sensitivity of Optimum Solutions to Problem Parameters*, AIAA 81-0548R, AIAA Journal, Volume 20, Number 9, September 1982, 1291-1299.

APPENDIX A

CONTENTS

FIGURES

THE ROLE OF OPTIMIZATION IN EMERGING COMPUTING ENVIRONMENTS FOR LIFE CYCLE ENGINEERING

This appendix explores the relationship between life cycle engineering methodology and advanced techniques for structuring and interpretation of computer programs, beginning with the architecture and integration requirements for a unified life cycle engineering (ULCE) design environment developed in Ref. A-1. The role of decision planning and methodology development in the life cycle engineering process is reviewed and implementation of a life cycle engineering environment using object-centered programming is outlined.

Advantages of the object-centered programming approach include the capability of high-level programming languages for design to represent design concepts in a computing environment, the use of instantiation to reduce the cost of generating design detail, and the suitability of constraint propagation as a technique for implementing a design methodology in a computing environment. The effect of high-level languages and instantiation on the cost of developing and using computing environments for design is discussed.

Constraint propagation is also defined and the reasons why it is appropriate as a tool for implementing a design process in an advanced computing environment are clarified. The relationship between meta-design and constraint propagation is presented; in an advanced computing environment for engineering design, the design methodology defines the computational agenda for constraint propagation.

To develop product designs balancing a range of life cycle requirements, existing techniques for constraint propagation must be extended to allow propagation of feasibility and optimality constraints. The techniques developed in this paper offer one way to accomplish this extension.

A. INTRODUCTION

Life cycle engineering involves an expanded scope in the requirements and trade-offs to be considered early in the design process. This expansion will result in increased system development costs. Early consideration of producibility and supportability requires a corresponding increase in the number of attributes and evaluation criteria. For example, instead of simply considering alternative configurations for a product, the elements of the support system and production process must also be considered. Clearly, some additional attributes are needed to describe alternative support and production concepts. It may also be necessary to define aspects of the configuration that are not used in the assessment of system performance to link the product concept to the production and support concepts. The number of alternative life cycle concepts, including system, production, and support concepts, that must be developed and evaluated to support technical decision-making increases in proportion to the number of attributes. Generation, definition, and evaluation of design alternatives are the primary sources of engineering costs in the early phases of design. Thus, life cycle engineering will require increased up-front investment.

Although funding levels for requirements definition and conceptual design are small in comparison with subsequent investments in product development, additional funding for concept exploration, concept definition, and demonstration/validation may simply not be available in a competitive economic situation. Even when funding is available, not all systems entering development ultimately reach operational capability. Thus, it may not be possible to justify additional up-front costs associated with life cycle engineering on the basis of potential savings in life-cycle costs. Better computing environments for design offer one way to reduce up-front design costs enough to make it economically feasible to apply life cycle engineering to a system development program without requiring a change in the funding profiles for system development programs.

B. META-DESIGN: DECISION PLANNING AND METHODOLOGY DEVELOPMENT IN THE LIFE CYCLE ENGINEERING PROCESS

The basis for meta-design is the principle that a convergent design decision-making sequence meeting all critical requirements is implicit in the technical description of the design concept. The idea is to formulate a design decision plan by extracting the design process from the design concept (Figure A-1).

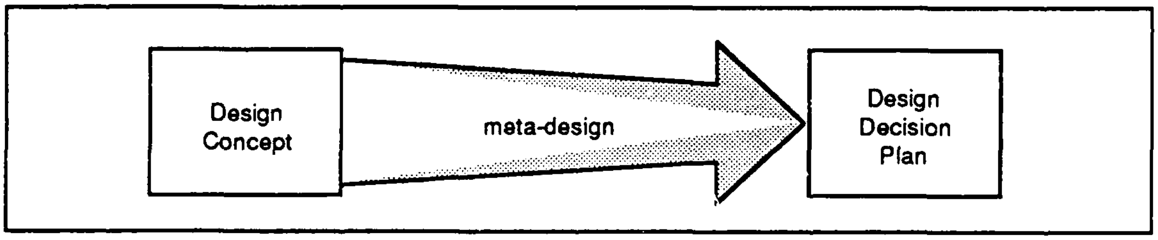


Figure A-1. Meta-Design Concept

The problem structure information is developed by the design team, which includes vendor representatives, manufacturing and support specialists, and specialists in various engineering disciplines such as (in the case of aircraft design) aerodynamicists, operations analysts, and structural analysts. This is done in the Generate Design Alternatives step of the life cycle engineering process (Figure A-2).

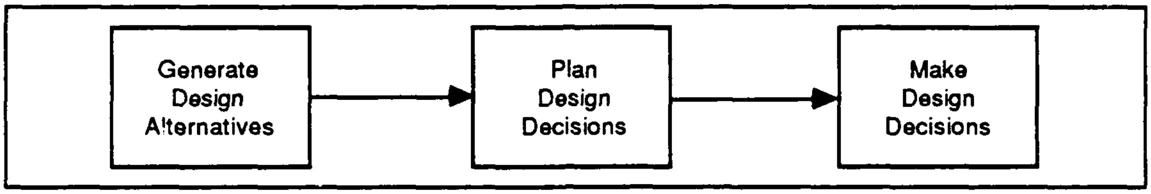


Figure A-2. Architecture of a Life Cycle Engineering Process.

Meta-design is accomplished in the Plan Design Decisions step, which takes place after design alternatives have been generated (and captured in a representation of the evolving life cycle concept, termed the Design in Progress) and before design decisions are made. To provide the information required for meta-design, a computing environment for life cycle engineering must provide explicit representation of system requirements, competitive strategies (goals), system functions, technical description of system design, production and support concepts, and engineering theories and models.

The Plan Design Decisions block of Figure A-2 is expanded in Figure A-3 to indicate how the design-in-progress information is used in the meta-design process. In describing the meta-design process, we first identify the elements of the life cycle engineering process architecture appearing in Figure A-3, and then proceed to discuss the process represented by the figure. The technical content of the system life cycle concept is represented in the life cycle engineering environment by the Design in Progress. This technical content includes product definition, process (manufacturing) definition, and definition of support concepts. The Design-in-Progress is comprised of three parts: the functional decomposition (what it does) , the system description (what it is), and the engineering theories and models (how it works). This partitioning is illustrated in Figure A-4. Requirements and goals are expressed in terms of the individual attributes within the functional decomposition and the system description. The engineering theories and models describe how system elements work together to accomplish specific functions. This information is developed iteratively in each pass through the Generate Design Alternatives step.

The explicit functional decomposition is essential for linking support, production, and vehicle or platform concepts. The functional decomposition includes both intended and unintended (failure) functions, to the extent that they are known by the development team. All technical concepts relevant to the life cycle, including product and process descriptions and support concepts, are in the system description. The engineering theories and models are quite broad in application, and include manufacturing process models, operational simulations, methods for evaluating human factors in maintenance operations, and analysis tools used in the more performance-oriented aerodynamics, structures, propulsion, and controls disciplines. Many of the engineering theories and models may require processes for using subjective judgments of experts to rank qualitative design alternatives for evaluation.

The life cycle engineering environment also includes information about the Design Decision Process. The life cycle engineering team create this process as part of the Plan Design Decisions step. This process provides the interfaces for controlling, tracking, and executing design decisions in the Make Design Decisions step of the life cycle engineering process.

To implement the Plan Design Decisions step, meta-design involves both identifying design decision elements and a sequencing these elements for execution. To identify design decision elements, the connectivity information contained in the Design in Progress is extracted. The nature of this connectivity information, for a simple aircraft design example, is indicated in Figure A-5. Engineering theories and models link the functional and system hierarchy decompositions.

Two important aspects of these connections are evident:

- A single engineering theory/model may connect several attributes of the functional and system hierarchy decompositions.
- Attributes of the functional and system hierarchy decompositions may be connected by more than one engineering theory/model.

These two features of the information contained in the Design-in-Progress representation connect the attributes of the system concept to those of the functional decomposition in a complex, graph-like topological structure. An example illustrating the basic concept is shown in Figure A-6. Here, the example of Figure A-5 has been extended to indicate the connections present in the Design-in-Progress before the selection of a power- or thrust-generating propulsive device has been made.

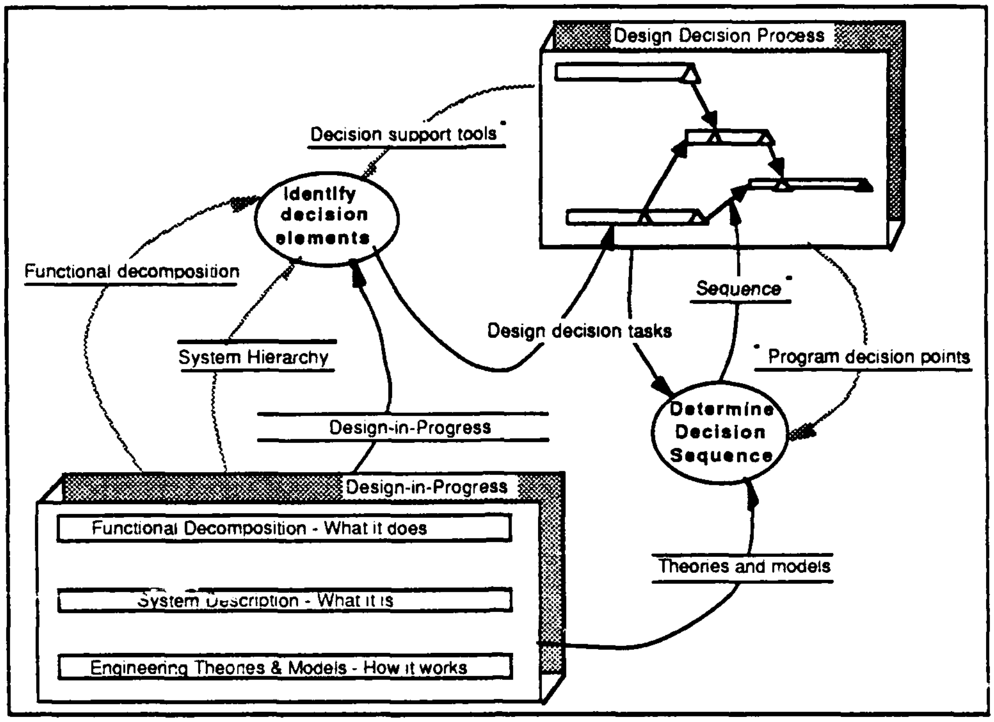


Figure A-3. Design Methodology Development and Design Decision Planning in the Life Cycle Engineering Process

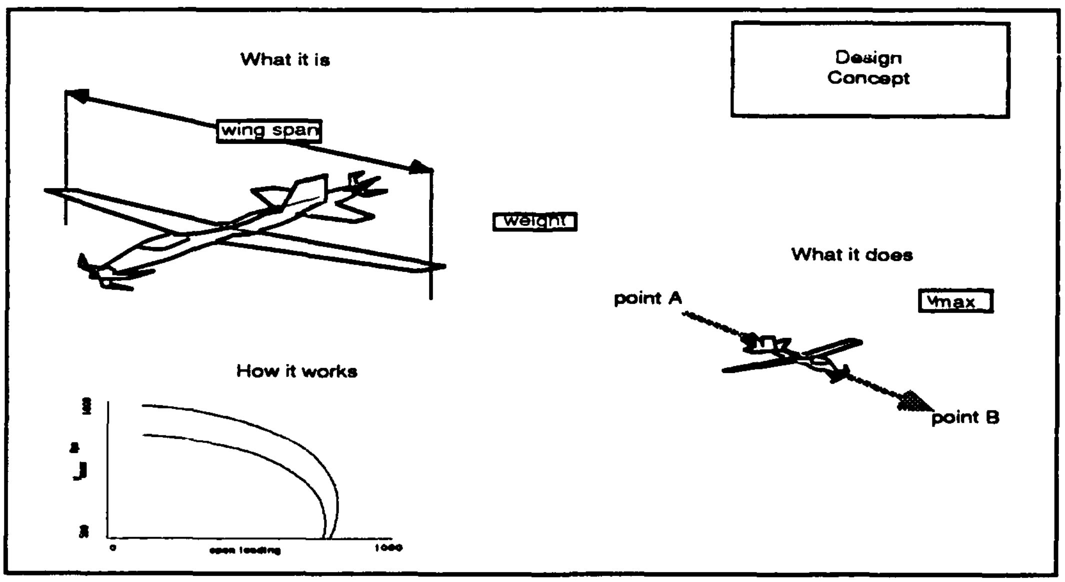


Figure A-4. Elements of a Design Concept

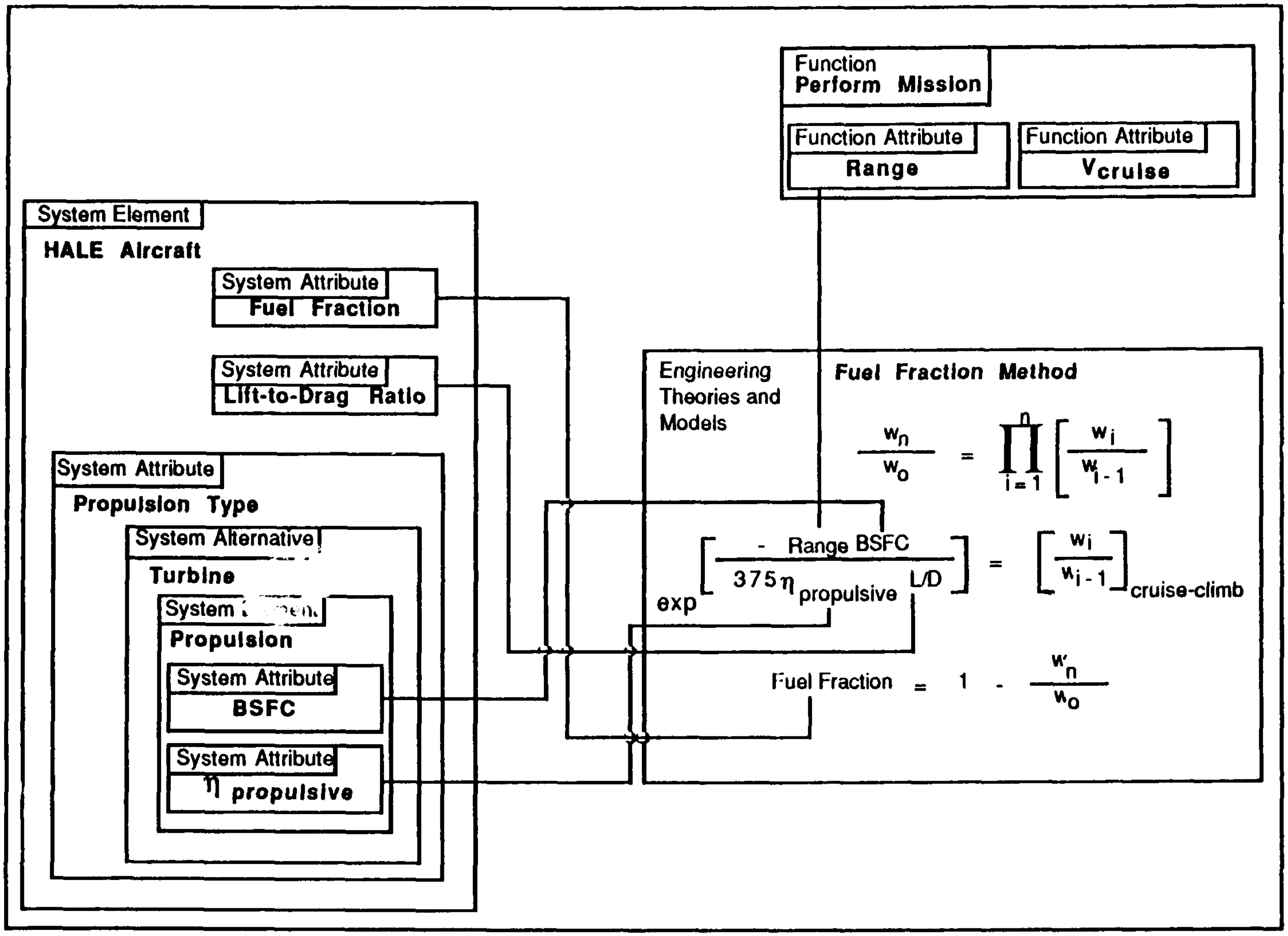


Figure A-5. Engineering Theories Link Functions and System Elements

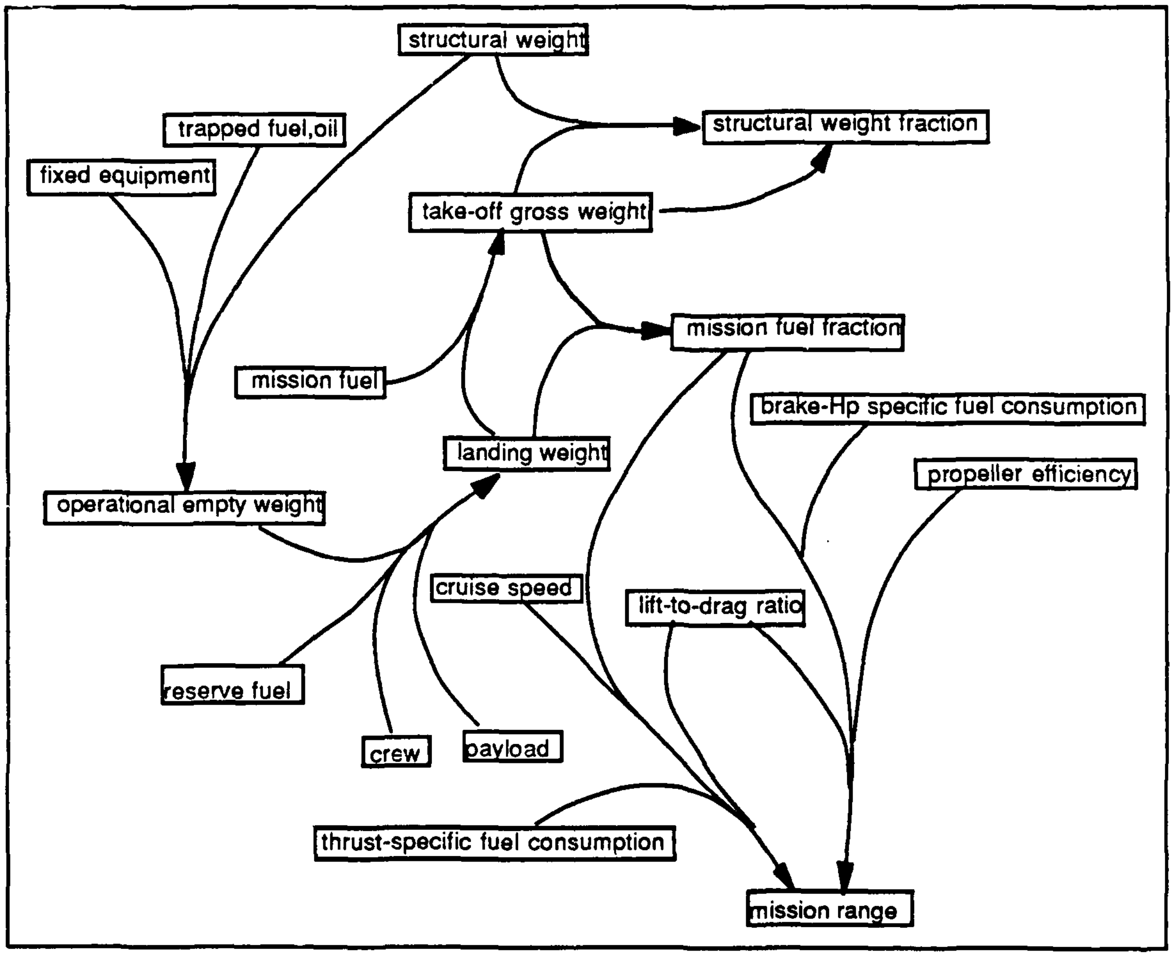


Figure A-6. G-digraph Representation of Design in Progress

The structure shown in Figure A-6 is not a graph in the usual sense (see Ref. A-2), for the relationships may connect more than two design attributes. Graphs represent binary relations (only two arguments). The Figure A-6 structure has been described as a generalized directed graph (G-digraph) [Ref. A-3]. The attributes of the life cycle concept are the vertices of the G-digraph, and the relationships among these attributes (implied by engineering theories and models) are represented by directed edges (in the sense of Ref. A-3) connecting these vertices.

Design decision elements are identified by combining attributes (vertices of the G-digraph) that must be considered together to make a design decision. Design decision elements would normally be arranged to fit into the structure of the system hierarchy and subsequently the work breakdown structure. (The relationships among design decisions are not always hierarchical, however). Individual decision elements must also be compatible with available decision support tools, as indicated in Figure A-3. The resulting design decision tasks become part of the Design Decision Process.

Once the decision elements have been identified, a sequence for making these decisions is determined. The sequencing of the decisions must result in a feasible, balanced, "optimal" design with a minimum of iteration. In addition to the connectivity information, considerations such as monotonicity and sensitivity, essentially analytical in nature, are important from the point of view of converging the design iterations. This information is available in the engineering theories and models of the Design in Progress. (Examples illustrating the use of analytical information about relationships implied by engineering theories and models to sequence design decisions are found in Chapter IV and Appendices C and D of this paper.) The decision sequence must be arranged to provide the information needed to support specific program decision points. This sequencing information is included, along with the definition of specific decision elements, in the Design Decision Process element of the life cycle engineering process architecture.

Design decisions are executed in the Make Design Decisions step. Making these decisions establishes specifications for system attributes and defines requirements for the next level of the ULCE process. Defining system functions to meet the requirements emerging from the design decisions initiates a Generate Design Alternatives step at the next level of detail in the definition of the system life cycle concept. The life cycle engineering process iterates through levels of design decisions in this way until sufficient technical information has been developed to support the relevant program management (and subsequently operational) decision.

We now consider an approach to implementing a computing environment capable of supporting this process architecture for life cycle engineering. This approach is based on the application of advanced computing technology.

C. EFFECT OF EMERGING COMPUTING TECHNOLOGY ON LIFE CYCLE ENGINEERING

Advanced techniques for organizing and using computer programs make a design computing environment for life cycle engineering feasible in terms of the associated up-front investment costs. The strategy is to reduce these costs by using abstraction to manage complexity. Although procedures and data are usefully abstracted in design software, perhaps the most important application of the advanced computer programming techniques is in the use of metalinguistic abstraction and modularity, objects, and state [Ref. A-4] to represent design concepts and processes. The computer programming techniques discussed in this section -- object-centered programming, instantiation, and constraint propagation -- are applications of these ideas.

1. Object-Centered Programming and Instantiation

Object-centered programming is a strategy for designing computer programs in which the structure of the program is based directly on the structure of the system being modeled [Ref. A-4]. To illustrate the idea, consider an object-centered system for the conceptual design of aircraft. The developer of the system organizes the computer program using wing, fuselage, lifting surface, mission, and other objects, instead of programs, subroutines, and functions. The immediate advantage in terms of productivity of the product development team is that translation step (from concepts relevant to the engineering problem to computer programming language constructs) is greatly reduced, if not eliminated altogether. A representation of the engineering problem in software that corresponds directly to the structure of the problem (as an engineer understands the problem) considerably enhances computer program readability and reduces the number of obvious programming blunders.

Additional advantages benefit the developer/engineer. In a computing environment that supports the object-centered programming strategy, modularity is strongly enforced within the objects. Thus, entire FORTRAN programs can reside within an object that controls their execution. Procedures hidden within objects are sometimes called methods for the object. Data can also be stored in objects using local variable names. Extension of the capability of a design system implemented using object-centered programming is considerably simplified because the developer can program at a very high level without considering the operating system or memory management.

Inheritance is often supported in such a computing environment. Using inheritance, the developer can represent concepts common to many objects using a single object. Tools used by many objects, such as icons or windows, are often implemented as objects whose properties can be inherited by another object. This idea can also be applied to technical constructs, such as point-in-space or lofted surface.

Instantiation is usually considered to be an essential part of an object-centered programming system -- the object defined in software by the developer is a kind of template or prototype for individual instances of that object. Instances of an object would normally be created by the user. Thus, the developer of design software would define a wing object, representing the definition of the wing technical concept in software, and designer/users of the system would define alternative wing designs by making instances of the wing object.

The local instance variables defined for an object can have different values bound to them in different instances of the object. Thus, wings with different aspect ratios, wing spans, airfoil sections, or aft spar heat treatment processes can be defined and managed as the design progresses.

Instances of objects interact with each other and with the user by passing messages. Examples of messages include requests from the user to obtain or set the value of a local variable or the user or another instance of some object instructing an instance to execute one of its local methods.

Some experience in the development and application of these tools has been gained in the research projects described in References A-5, A-6, A-7, A-8 and A-9. Based on this experience, informal estimates of the expected improvements in designer/user productivity have been made. Most of the estimates of increases in productivity range between a factor of 4 and 12, although increases in designer/user productivity as great as a factor of 40 have been estimated. The Automated Airframe Assembly Program (AAAP), currently being performed by Northrop for USAF Materials Laboratory, has provided preliminary examples of the designer/user productivity increases, relative to current state-of-the art CAD (computer-aided design) systems, that can be achieved in detail design and manufacturing engineering. Improvement of designer/user productivity, in terms of the time to define a model, by a factor slightly greater than 30 is seen in the results presented in Ref. A-9. Two conclusions can be drawn immediately from these preliminary estimates:

- Additional research is needed to obtain and substantiate estimates of the effect of advanced computing technologies on designer/user productivity for various types of design problems, and
- Computing technology is emerging that developers of design tools, and subsequently designers, can use to increase design productivity enough to offset the considerable increase in the complexity of the design problem associated with up-front consideration of downstream producibility and supportability concerns.

2. Object-Centered Implementation of an ULCE Environment

The Design-in-Process and Design Decision Process elements of the Reference A-1 architecture for an ULCE environment can be implemented using an object-centered programming approach. The advantages of such an implementation are described in the following paragraphs.

The information content of the functional decomposition has a relatively simple structure, which can be implemented using conventional programming methods, such as relational database management systems. However, a description of the functional breakdown using system function objects would simplify the development of user interfaces. Good user interfaces are essential to make the coupled, leveled structure of the functional decomposition accessible to members of the product development team.

Object-centered techniques would have considerable value for implementing the system description component of the Design-in-Progress. Here, system element objects could be defined, with local variables and methods providing access to geometry, material and process specifications, technical orders, and other views of the objects. Manufacturing processes and support concepts are also represented in the Design in Progress. This capability then provides the product development team (including producibility and supportability engineers) with an environment to define alternative manufacturing plans and maintenance procedures and to evaluate them along with system alternatives.

A multilevel approach to defining and instantiating system element objects would provide the design team with the capability to partially instantiate designs for evaluation and comparison. The status of instances of design alternatives could be managed using local state variables. By virtually eliminating the cost of generating design detail, instantiation can make system-level trade studies, and the capability to adapt the product or process to respond to them, a reality throughout the product life cycle.

The representation of engineering theories and models using object-centered programming techniques also offers significant benefits. The capability of objects to manage the execution of computer programs written in FORTRAN and other languages would be quite valuable. Explicit representation (as objects) of alternative theories and models describing the accomplishment of the same group of system functions at different levels of system description definition precisely represents the idea of levels of approximation. This information is extremely useful in managing design state and assessing levels of risk throughout the design process.

Finally, design decision objects could be defined in the Design Decision Process element of the ULCE architecture. The meta-design process would result in the instantiation of design decisions. During the Make Design Decisions step, the design decision objects provide user interfaces and manage the execution of procedures for decision support.

D. CONSTRAINT PROPAGATION AND META-DESIGN

1. Constraint Propagation

A constraint is a restriction on the values that may be taken on by a design attribute. (This usage is consistent with the concept of equality and inequality constraints in design optimization.) Equality and inequality constraints are here referred to collectively as feasibility constraints. Equality constraints may bind a design attribute to a specified constant, or they may bind several design attributes in an equation. Inequalities may also represent a relationship among several design attributes, or they may be restrictions on the values allowed for a single design attribute (in design optimization terminology,variable bounds).

Values that may be taken on an attribute may also be restricted by a competitive strategy or goal. Thus, the statement that a particular attribute is to be minimized, maximized, or close to a target value effectively constrains that attribute. The goal of balanced design also acts as a constraint in this sense, since balance is a form of optimality (Pareto-optimality). Restrictions of this type are also considered here to be constraints (optimality constraints).

An extension of existing methods for propagating equality constraints to handle optimality and feasibility constraints can be based on the methods used in this paper. To illustrate the idea, we first consider propagation of equality constraints. (Propagation of feasibility constraints, of a single optimality constraint, and of multiple optimality constraints are considered in Appendices B, C, and D of this paper.)

Constraint propagation is a technique for structuring computer programs to represent constraints directly in software. The distinction between the definition of aspect ratio,

$$\text{aspect\_ratio} = \text{span}^2/\text{area}$$

and an instruction in a conventional computer program

```
aspect_ratio = span^2/area
```

or

set aspect_ratio to span^2/area

is essential for the concept. The instruction sets the value of "aspect_ratio" to the square of the value bound to the variable name "span," divided by the value bound to the variable name "area." The instruction is meaningless unless it can be executed. The definition of aspect ratio, on the other hand, can be interpreted as a declaration that aspect ratio is equal to the square of the span divided by the area. This declaration is independent of the values for span, area, and aspect ratio. More important, the definition of aspect ratio is a statement that, given values for any two of the three variables, we can solve for the other (we may have to specify which root to take).

In contrast, the instruction has meaning (it can be executed) only if values have already been bound to the span and the area variable names. Thus, the instruction implies a sequence of determining the design attributes . The definition does not. The programming style in which constraint definitions are represented explicitly in the software is often called declarative. The conventional approach, working directly with instructions, is described as imperative. This distinction extends beyond programming style, and can be used to characterize declarative and imperative types of knowledge, as is done in Ref. A-4:

> "The contrast between function and procedure is a reflection of the general distinction between describing properties of things and describing how to do things, or, as it is sometimes referred to, the distinction between declarative knowledge and imperative knowledge."
>
> ". . . an important current area in programming-language design is the exploration of so-called very high-level languages, in which one actually programs in terms of declarative statements. The idea is to make interpreters sophisticated enough so that, given "what is" knowledge specified by the programmer, they can generate "how to" knowledge automatically. This cannot be done in general, but there are important areas where progress has been made."

The sequence of design decisions is to be identified through meta-design in the life cycle engineering architecture. Conventional computer programming, using instructions to represent design concepts, severely limits this flexibility. Thus, to use meta-design to advantage, we should use declarative programming to represent system life cycle concepts.

When the Make Design Decisions step of the Figure A-2 life cycle engineering process architecture is executed, we will need a procedure for translating the flexible declarative description of the design concept into a set of instructions for binding the design attributes to specific choices. How this can be done using an object-centered implementation of constraint propagation is detailed in the following paragraphs.

In an object-centered system for constraint propagation, the constraints themselves are represented as objects. The defining equation of the constraint is an instance variable. This allows different instances of the constraint object to represent different constraints. In one approach to constraint propagation, the fixed point method of Elias [Ref. A-5], the constraint objects have a method for using the values bound to all but one of the variables to solve for the remaining one.

Instances of the constraint objects must also manage information about the defining equations. As an example, it is quite important to know that aspect ratio cannot be determined from the span and the area when the area is zero, or that the determination of span from aspect ratio and area is not unique (both b and -b are solutions).

The variables appearing in the defining equations represented by the constraint instances are also implemented as objects. The purpose of the defining equation variable objects is to manage the state of the design. The state of an instance of a defining equation variable object is represented by an instance variable that is given a value "user-specified," "guess," or "computed." Figure A-7 depicts constraint and variable objects implementing design attributes and relationships for the aircraft aircraft sizing problem shown in Figure A-6.

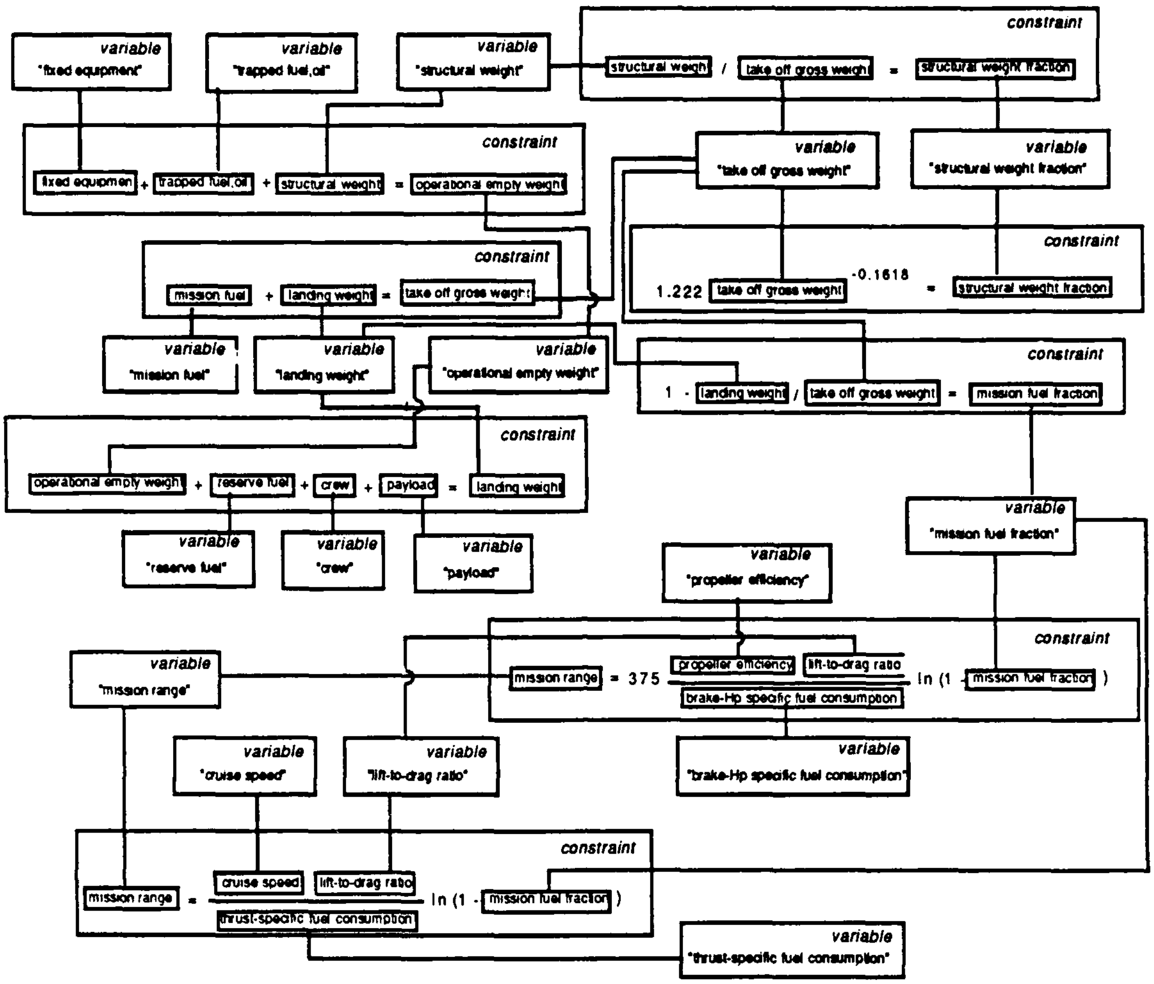


Figure A-7. Constraint and Variable Objects

A computer program employing constraint propagation is used in the following way. The user specifies fixed values for some of the design attributes. The program then uses information about the relationships (whether a defining equation can be used to solve for the set of variables that has been specified) to develop a computational agenda [Ref. A-5]. The computational agenda is a sequence of steps (a design path) for using the fixed values and the available relationships to solve for the unspecified values. The computational agenda can then be interpreted as instructions and executed as a computer program.

All combinations of user-specified initial values can be propagated. Kolb [Ref. A-11] presents a detailed investigation into the complications that may result when a relationship with a multivalued inverse is reversed in the design path.

The transition of defining equation variable states from guess to computed seems to propagate across the G-digraph as the computational agenda is executed, hence the term constraint propagation .

Elias [Ref. A-5] recognized the close analogy between the choice of which attributes to bind to fixed values, development of a computational agenda, and solution of the constraint propagation network on one hand; and the requirements definition, meta-design, and sizing/synthesis elements of the aircraft design process on the other. In Elias' view, the computational agenda is the result of the meta-design step.

2. Relating Design Methodologies to the Propagation of Optimality and Feasibility Constraints

The constraint propagation idea is closely related to meta-design in a life cycle engineering environment. Advanced computing technologies will be needed to implement a life cycle engineering architecture in a competitive economic climate. Constraint propagation provides the means for the system development team to execute a design process represented in an advanced computing environment.

Inherent in the life cycle engineering concept is the idea of balanced design . Balancing a design against multiple criteria means that an improvement in any one criterion can only be obtained at the expense of worsening at least one of the other criteria. This concept is called Pareto-optimality .

In turn, Pareto-optimality is a parameterization of "ordinary" optimality by weighting factors representing the relative importance of the multiple criteria. Thus, to address life cycle engineering in an advanced computing environment for design, we need to understand how to propagate optimality and feasibility constraints.

We construct a computational agenda for constraint propagation from the decision sequence identified using methods such as those of Chapter IV of this paper. Such a design process can then be executed by applying requirements to specify initial values for design attributes.

Decision elements are identified as groups of attributes of the design-in-progress that are to be determined together. One such grouping is shown in Figure A-8.

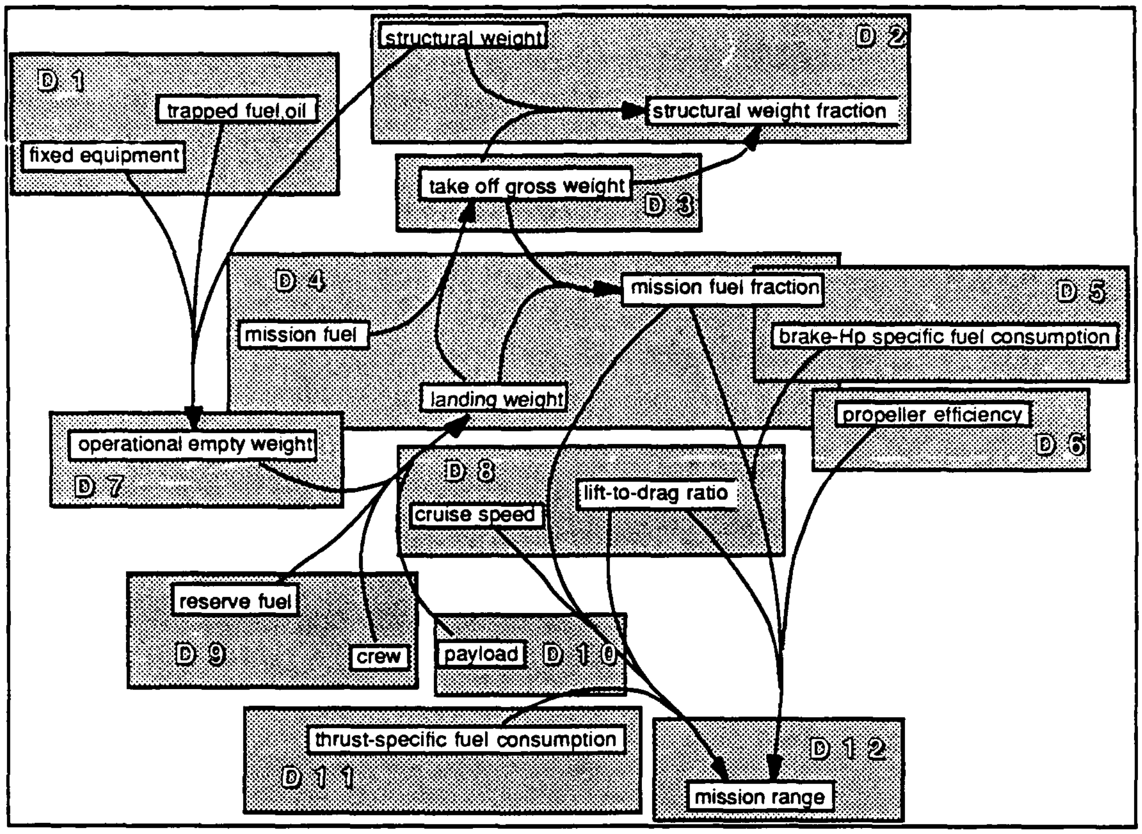


Figure A-8. Identification of Decision Elements

Making design decisions results in the determination of values for the design attributes. Since the attributes are related, the decision elements are also related. Thus, the complex G-digraph structure shown in Figure A-6 induces an ordinary directed graph structure on the decision elements in Figure A-8, as shown in Figure A-9. The edges on the Figure A-9 directed graph indicate, for example, that determining take-off gross weight by making decision D3 will have a direct effect on decision D2. This reflects the fact that there are two directed edges from D3 to D2 in the G-digraph shown in Figure A-6. The directed edge connecting decision D4 to decision D3 indicates that D3 will be affected by the values for landing weight and mission fuel selected in decision D4.

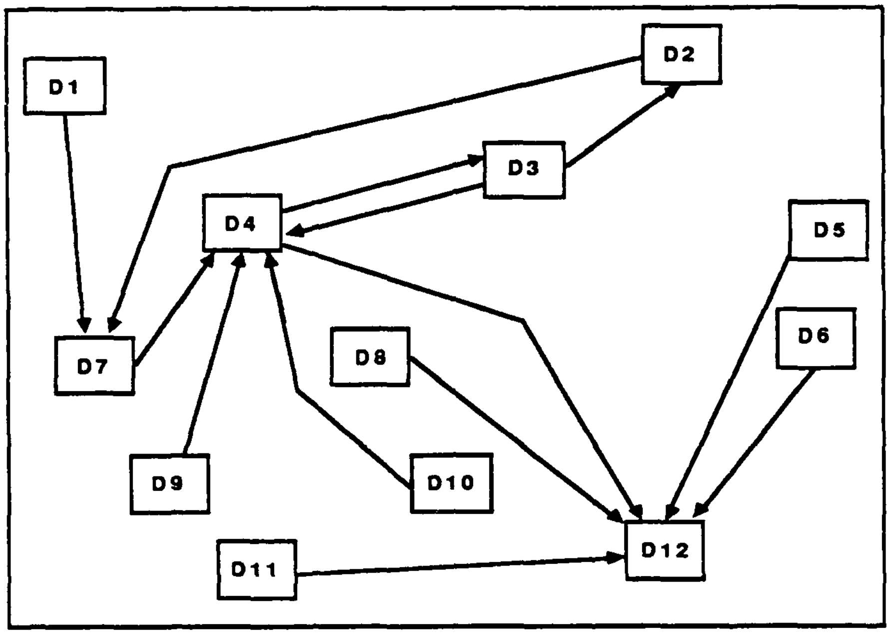


Figure A-9. Relationships Among Decision Elements

As illustrated in Figure A-10, sequencing of the decision elements is accomplished by eliminating some of the edges from the directed graph of Figure A-9.

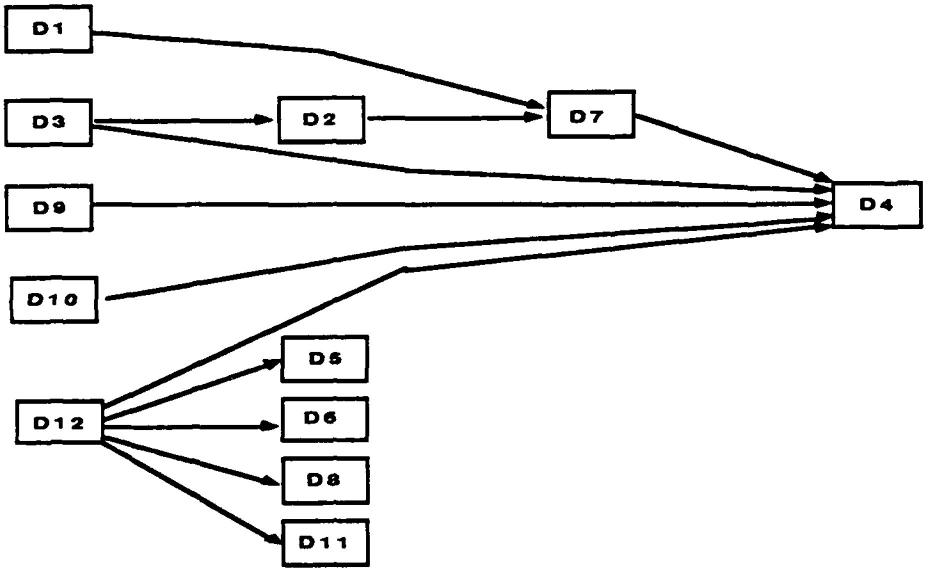


Figure A-10. Decision Sequence

Given a decision sequence, a computational agenda can be constructed as shown in Figure A-11.

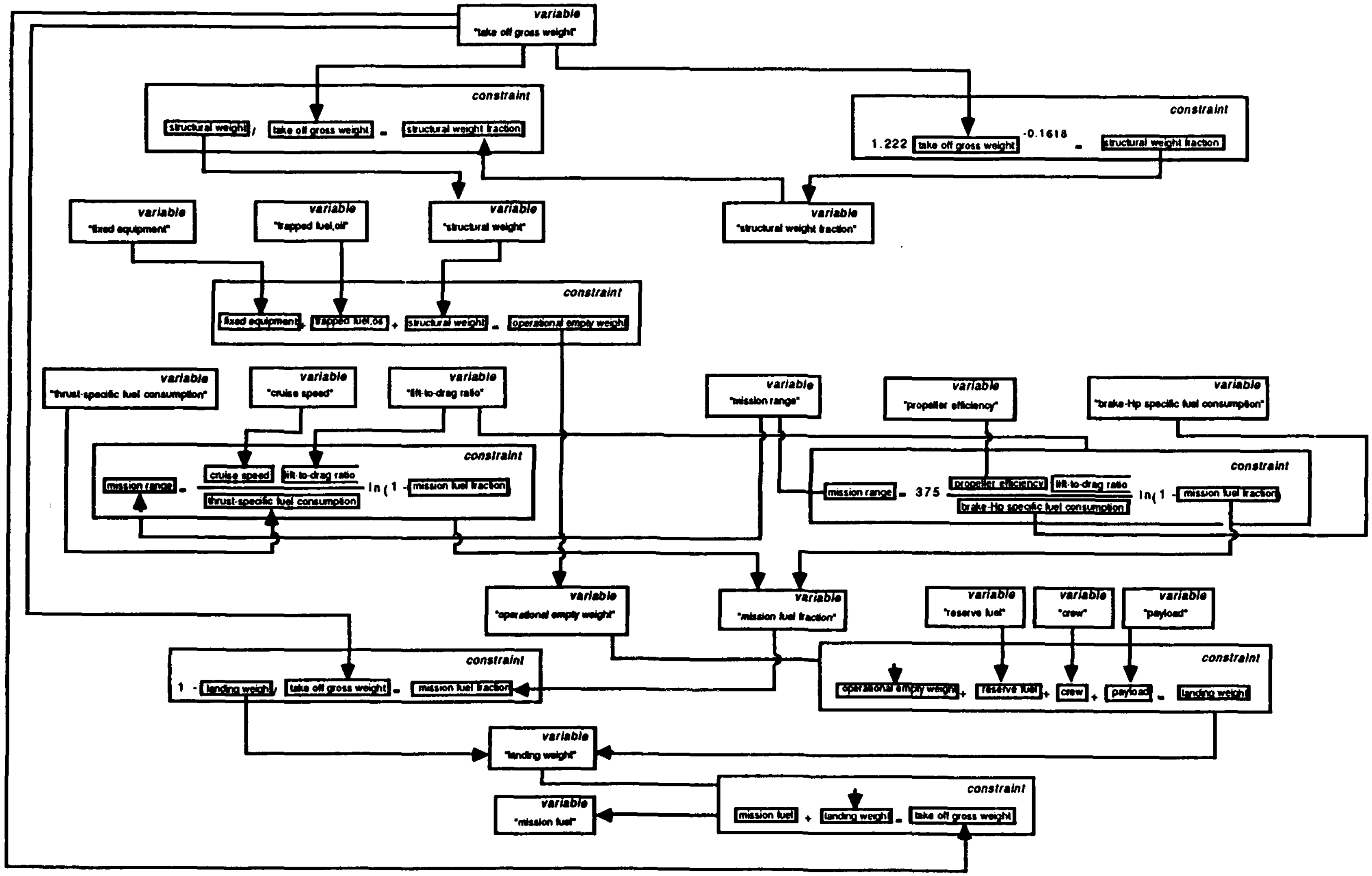


Figure A-11. Computational Agenda for Constraint Propagation.

To build a computational agenda, declarative information about the design concept is interpreted as a procedure for setting values for the design attributes. The sequence of imperative steps in the procedure is based on the decision sequence in Figure A-10. The initial selections of values for design attributes are made in decisions D1, D3, D9, D10, and D12. For example,take-off gross weight is given a value in decision D3. Once take-off gross weight has been determined, we can apply the constraint

$$\text{structural weight fraction} = 1.222\ (\text{take-off gross weight})^{-.1618}$$

to determine a value for the structural weight fraction . Using the take-off gross weight and structural weight fraction , the structural weight can be found by inverting the constraint

$$\text{structural weight fraction} = \text{structural weight} / \text{take-off gross weight},$$

as indicated by the arrows in Figure A-11.

Continuing along these lines, the constraints imposed by selecting values for the trapped fuel and oil , fixed equipment, reserve fuel weight , payload weight , crew weight , and mission range are propagated to determine values for all of the other design attributes. Since the computational agenda is a procedure for determining these values, it can be interpreted and executed as a computer program.

The relationship between constraint propagation and requirements flowdown can now be clarified. Design attributes such as mission range and reserve fuel are typically set by requirements (such as those specified in the RFP, MIL-Specs, or FARs). The flowdown of these requirements to specification of system attributes is clearly traced by the computational agenda. A similar process takes place as the system development team makes decisions that place constraints on the choices available to subsystem development teams.

A computational agenda can be constructed from any decision sequence. A typical design goal in conceptual sizing is to minimize take-off gross weight. The decision sequence in Figure A-10 is neither optimizing or feasible. Since the take-off gross weight is set initially, a single pass through the decision sequence cannot make any progress toward an optimal value. The decision sequence is not feasible, either. Tracing the sequence of computations indicated in Figure A-11 will show that the mission fuel fraction and the landing weight appear to be over-determined. In fact, the intent of decision D4 is to compare the two values for the mission fuel fraction to support the selection of a power- or thrust-generating propulsion subsystem. Thus, the mission fuel fraction is not really overdetermined. The landing weight is overdetermined, however. All of the constraints in which landing weight appears must be satisfied identically. An example of a balanced design methodology for an aircraft sizing problem is presented in Appendix D.

In Appendix B, we present and analyze an effective procedure for structuring a decision sequence. We prove that this procedure ensures that execution of the computational agenda will make progress toward optimization of a single design objective function, or balance of multiple objective functions, while maintaining a feasible design. Since constraint propagation is effected by execution of a computational agenda, this procedure can be usefully considered to be a method for the propagation of feasibility constraints, and a single optimality constraint, or balanced design constraints.

REFERENCES

A-1 Brei, M.L., et al; *Architecture and Integration Requirements for an ULCE Design Environment*, IDA Paper P-2063, April 1988.

A-2 Carré, Bernard, *Graphs and Networks*, Clarendon Press, Oxford, UK, 1979.

A-3 Tang, W. Michel, A.N.; and Hale, H.W., "On Structure and Stability of Interconnected Dynamical Systems", IEEE Trans. on Circuits and Systems, v. CAS-27, no. 5, May 1980.

A-4 Abelson, Harold and Sussman, Gerald Jay with Julie Sussman, *Structure and Interpretation of Computer Programs*, The MIT Press, Cambridge, MA; McGraw-Hill, New York, 1985.

A-5 Elias, Antonio L., "Knowledge Engineering of the Aircraft Design Process",Chapter 6 in *Knowledge Based Problem Solving*, Kowalic, J.S., ed., Prentice-Hall, Englewood Cliffs, New Jersey, 1985.

A-6 Bouchard, E.E.; Rogan, J.E.; and Kidwell, G., *The Application of Artificial Intelligence Technology to Aeronautical System Design*, AIAA-88-4426, AIAA/AHS/ASEE Aircraft Design, Systems, and Operations Meeting, Atlanta, GA, Sept. 7-9, 1988.

A-7 Kolb, M.A., *A Flexible Aid for Conceptual Design based on Constraint Propagation and Component-Modelling*, AIAA-88-4427, AIAA/AHS/ASEE Aircraft Design, Systems, and Operations Meeting, Atlanta, GA, Sept. 7-9, 1988.

A-8 Rosenfeld, L., *Intelligent CAD Systems - CAEDM for the 90's*, 6th National Conference on University Programs in Computer-Aided Engineering, Design, and Manufacturing, Atlanta, GA, June 1988.

A-9 Mayhew, Bruce; Hitchock, Michael, et al: *Georgia Tech/Automated Airframe Assembly Program Technology Transfer Meeting*, Georgia Institute of Technology, Atlanta, GA, October 25-27, 1989.

A-10 Justice, R.S.; Harcrow, A.L.; and Isurietta, C.E., *SGASP Version 3.0 Documentation*, Vols. 1 (User Guide) and 2 (Developer Guide), Engineering Report LG87ER0171, Lockheed Aeronautical Systems Co., October 1987.

A-11 Kolb, M.A., "On the Numerical Solution of Simultaneous, Non-Linear Equations in Computer-Aided Preliminary Design", FTL Report R86-1, Flight Transportation Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1986.

APPENDIX B

CONTENTS

PROGRESS TOWARD A THEORY OF OPTIMAL DECISION SEQUENCES

This appendix provides the technical justification for the application of optimization theory to evaluate design methodologies. Here, we prove the following results:

- Global convergence of a design methodology follows from global convergence of the methods used to solve the subproblems (decision elements). Here global convergence means that the algorithm converges from any initial design point [Ref. B-1].
- The convergence test $df/dx_i = 0$ (df/dx_i is an optimal sensitivity derivative), for all i, is satisfied at, and only at, a Kuhn-Tucker-Karush point.
- $dF/dp\ (\omega) = \Sigma\ \omega_i\, df_i/dp$

where $F = \Sigma\ \omega_i f_i$. The last equality expresses the form of the relationship between the optimal sensitivity derivative of the objective function for a Pareto-optimization problem and the optimal sensitivity derivatives of the multiple objectives. We establish that this equation also describes the effect on the Pareto-optimal sensitivity derivative dF/dp of changes in the relative prioritizations ω_i.

These results provide the justification for the technique of using approximations to the objective functions of subproblems as the optimality conditions for a Pareto-optimization problem. This technique forms the basis for methodology F of Chapter IV and for the solution of a Pareto-optimal aircraft sizing problem in Appendix D.

The precise relationship of design methodologies structured using the procedure described in this paper to other decomposition methods of design optimization has not yet been established. The point of view of the method of this paper, the method itself, and a fortiori , the convergence results of this appendix, are essentially new. However, the method of this paper is closely related to the techniques described in reference B-2 and work cited there. The method differs from coordinate descent methods, such as that of Gauss-Southwell (discussed in reference B-1), in that the optimization subproblems may be constrained and may include several design variables.

A. AN OPTIMIZING PROCEDURE FOR SEQUENCING DESIGN DECISIONS

Planning a design decision making process requires a procedure that can be used to identify and sequence design decision elements. In the context of optimization theory, an effective procedure for design decision making is a technique for iterative solution of an optimization problem, P. This problem can be stated as follows:

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{g}(\mathbf{x}) \leq \mathbf{0} \\ & \mathbf{h}(\mathbf{x}) = \mathbf{0} \\ & \mathbf{x}_{\text{lower}} \leq \mathbf{x} \leq \mathbf{x}_{\text{upper}} \end{aligned}$$

where f is a scalar-valued function of a vector variable

$$\mathbf{x} = (x_1, x_2, \ldots, x_n).$$

f is referred to as the objective function . The x_i 's are the design variables, and **g** and **h** are vector-valued functions of **x**. The components of **g** are the inequality constraints

$$g_k(x_1, x_2, \ldots, x_n) \leq 0 \qquad k = 1, \ldots, n_i$$

and the components of **h** are the equality constraints

$$h_l(x_1, x_2, \ldots, x_n) = 0 \qquad l = 1, \ldots, n_e$$

$\mathbf{x}_{\text{lower}}$ and $\mathbf{x}_{\text{upper}}$ are vectors of upper and lower bounds for **x**. A starting value for the design variable vector, $\mathbf{x}^0$, satisfying the upper and lower bound inequalities, is given.

P is decomposed into subproblems, referred to as decision elements. In this decomposition, each decision element D_j is associated with a nonempty subset of the x_i 's. Each x_i is assigned to a unique D_j. The x_i 's assigned to D_j are called local design variables for D_j .

Each decision element D_j is also associated with an optimization subproblem or a feasibility subproblem. These subproblems include all of the constraints g_k and h_l in which any local design variable for D_j is explicit. If a local design variable x_i is explicit in the objective function f, then the subproblem is formulated as an optimization subproblem with f as objective function. If not, D_j is a feasibility subproblem. Design variables x_i not assigned to D_j may be explicit in the constraints or the objective function of the optimization/feasibility subproblem associated with D_j. These design variables are parameters for D_j. Each design variable of the problem P may be a parameter for a number of decision elements.

Two decision elements may be combined by forming the union of their local variable sets and formulating a new optimization or feasibility problem in the obvious way.

We now drop the distinction between the decision element D_j and its associated optimization or feasibility subproblem.

The values of all parameters are fixed during the solution of D_j. In a sequential solution process, the values for the local design variables determined by the solution of D_j are passed as parameters to subsequent decision elements.

The term solution sequence is used here to mean a directed graph with the decision elements as nodes. The edges are labelled with a vector of bindings for the parameters. The directed graph represents the order in which the decision elements are to be executed or solved. This definition allows us to consider multiple predecessors and successors, as well as concurrent solution of decision elements.

Convergence of the design decision plan to an optimal or balanced design is ensured by placing restrictions on the allowable sequences of decision elements. It may be necessary to combine decision elements to construct such a solution sequence. If decision elements may be combined, the set **S** of optimal and feasible solution sequences is not empty if there is a procedure for finding an optimal feasible solution for P. The solution sequence $\{P\}$ is an element of **S**, since P can be recovered by combining all of the decision elements.

1. Feasible and Optimal Decision Sequences

A design decision D_1 can be made before another design decision D_2 if the values chosen for the design attributes in D_1 do not make D_2 infeasible. For example, say x_1 is a design variable to be determined by D_1 and x_2 a design variable to be determined in decision D_2. Say x_1 and x_2 are coupled by an inequality constraint g :

$$g(x_1, x_2, \ldots) \leq 0 .$$

In sequencing D_1 and D_2 we have three alternatives:

1) make decision D_1 before D_2; x_1 will then be fixed by D_1 and will be a parameter for decision D_2.

2) Make decision D_2 before D_1; x_2 will be a parameter in D_1.

3) Combine D_1 and D_2 into a single decision element.

Consider now the case where D_1 is sequenced before D_2. Solution of D_1 will result in a change Δx_1 from the initial value for x_1. The effect of this change on the inequality constraint g can be assessed with a first-order approximation:

$$\Delta g \sim (\partial g / \partial x_1)\, \Delta x_1$$

Thus if $(\partial g / \partial x_1)$ and Δx_1 are opposite in sign, Δg will be negative and g will be less critical in making decision D_2 (in comparison with the initial design). If $(\partial g / \partial x_1)$ and Δx_1 have the same sign, g will become more critical for D_2 if we make decision D_1 first.

Feasible sequences for the design decisions can be determined using the directions of proposed changes in the design variables in each decision and the signs of the partial derivatives of inequality constraints coupling two or more decisions together. The criteria are

F-1 If D_1 does not make (any of) the constraints of D_2 more critical, then D_1 can be sequenced before D_2.

F-2 If D_2 does not make (any of) the constraints of D_1 more critical, then D_2 can be sequenced before D_1.

If D_1 makes the constraints of D_2 more critical, and D_2 makes the constraints of D_1 more critical, then it may be necessary to combine D_1 and D_2 into a single decision element.

If both F-1 and F-2 are met, D_1 and D_2 can be made concurrently.

Many possible decision sequences may meet these criteria. In an extremely tightly coupled problem, all of the initial design decisions may be combined into a single design decision by this procedure. All of the decision sequences meeting criteria F-1 and F-2 will lead to feasible designs. We will next consider additional restrictions on the possible decision sequences, leading to an optimal, and subsequently, a Pareto-optimal or balanced design.

Determination of a sequence of design decisions leading to an optimal design requires an initial suboptimization pass through each of the decision elements. In this suboptimization pass, each of the decisions in which one of the objective functions for the design appears explicitly as a function of the local decision variables is solved in isolation. The results of the suboptimization pass are then analyzed using sensitivity of optimal solutions to problem parameters. That analysis is used to establish whether an iteration of the decision-making procedure will progress toward an optimal design.

In constructing a decision sequence leading to an optimal design, we again have the three alternatives: place D_1 before D_2 in the decision-making sequence, place D_2 before D_1, or combine them. Let $f(x_1, x_2, \ldots)$ be an objective function to be minimized in both D_1 and D_2. If D_1 is made before D_2, then x_1 appears in D_2 as a parameter. The sensitivity of the optimal solution to D_2 to the parameter x_1 is df/dx_1. We know the directions of proposed changes in the design variables (from the suboptimization pass), so we can determine

$$\Delta f \sim (df/dx_1)\,\Delta x_1$$

Thus, if df/dx_1 and Δx_1 are opposite in sign, Δf will be negative. Then if D_1 is made before D_2, f will not increase during the decision subsequence $\{D_1, D_2\}$. Any decision subsequence in which f will not increase can form part of an optimizing decision sequence. Optimizing decision sequences are built up from such subsequences, with one additional criterion: decision elements with $df/dx_i = 0$ must be placed after decision elements with $df/dx_i \neq 0$. The need for this criterion will emerge from consideration of convergence questions.

We now establish global convergence of an effective procedure for solving an optimization problem using an optimizing sequence of design decisions.

2. An Effective Procedure for Optimization through the Sequence of Design Decisions

Step 0. Initialization. Choose an initial design within the variable bounds and make an initial choice of decision elements.

Step 1. Evaluate each decision element to determine an optimum solution for that decision element (in isolation).

Step 2. Identify possible feasible decision sequences. If feasibility requires combination of decision elements, iterate with Step 1.

Step 3. Identify possible optimal decision sequences. Check convergence. If solution is converged, stop. If optimality requires combination of decision elements, iterate with Steps 1 and 2.

Convergence criterion: Both

(i) design variables did not change during last solution pass and

(ii) all optimal sensitivities are zero ($df/dx_i = 0$ for all parameters x_i) must be satisfied.

Step 4. Select a decision-making sequence that is both feasible and optimal. If D_i is sequenced before D_j, the number of parameters passed from D_i to

D_j must equal or exceed the number of independent active constraints common to both decision elements.

Step 5. Find an optimal solution for each decision element in sequence. Update the values of all design variables and iterate from Step 2.

3. Proof of Convergence

We would like the procedure to converge to an optimal solution from any initial design within the variable bounds. This property is known as global convergence in optimization theory [Ref B-1]. (Global convergence is not the same as convergence to a global optimum.)

In the theory of convergence, an algorithm is a mapping, often considered to be a point-to-set mapping, although that level of generality is not needed to analyze the convergence of the procedure presented here. An algorithm corresponding to a feasible, optimizing decision sequence can be thought of as a vector-valued function, **A**, of a vector variable **x**. **x** is the vector of design variables $(x_1, x_2, \ldots)$. A particular value of **x**, corresponding to choices for each of the design variables, is called a design. **A** maps the design before an iteration of Steps 2 through 5 (above), that is, $\mathbf{x}_i$, to a new design determined by the algorithm, $\mathbf{x}_{i+1} = \mathbf{A}(\mathbf{x}_i)$. A sequence of points $\{\mathbf{x}_i\}$ obtained by iteration of the algorithm in this way is said to be generated by the algorithm **A**.

Global convergence for the algorithm **A** follows from the following conditions [Ref. B-1]:

1. Sequences generated by **A** remain in a compact (closed, bounded) set.
2. The mapping defined by **A** is continuous.
3. A descent function can be defined for **A**.

Condition (1) follows directly from global convergence of the algorithms used to optimize individual decision elements.

Condition (2) will follow from continuity of the objective and constraint functions if the optimal sensitivity derivatives are continuous . The optimal sensitivity derivatives may have discontinuities at values of the parameter corresponding to changes in the active constraint set. These discontinuities can be removed by replacing the optimal sensitivity derivative with a continuous approximation. A suitable procedure (a closed algorithm choosing a feasible, optimal decision sequence must also be specified.

Condition (3) requires us to find a descent function for **A**. A descent function measures the progress of the algorithm toward a solution. The descent function must decrease with each iteration until the solution set is reached. A penalty function, P, formed from the problem objective function and constraints, will be a descent function for A provided that P is a descent function for the globally convergent algorithms used to optimize each decision element and decision elements with $df/dx_i = 0$ are be placed after decision elements with $df/dx_i < 0$. Then the first decision elements in the sequence will decrease P unless all of the df/dx_i are 0. Subsequent decision elements will not increase P. Then P will decrease with each iteration until a local optimum for the whole problem is reached.

Thus the solution set for the algorithm is defined by the condition that all $df/dx_i = 0$. We must now show that this solution set is the set of Kuhn-Tucker-Karush points.

A Kuhn-Tucker-Karush point is a point satisfying the Kuhn-Tucker-Karush optimality conditions:

1. Feasibility

$$\mathbf{g}(\mathbf{x}) \leq \mathbf{0}$$

$$\mathbf{h}(\mathbf{x}) = \mathbf{0}$$

2. Active constraints:

$$\lambda_k \geq 0 \text{ and } \lambda_k g_k(\mathbf{x}) = 0 \quad k = 1, \ldots, n_i$$

3. Extremum of the Lagrangian over the primal space:

$$\partial f/\partial x_i + \Sigma \lambda_k \partial g_k/\partial x_i = 0 \qquad \text{for each design variable } i.$$

Convergence of the procedure for optimization through decision sequencing to a Kuhn-Tucker-Karush point follows from

Proposition 1: Let $\mathbf{x}^*$ be a point generated by the decision sequence. If all df/dx_i are 0 at $\mathbf{x}^*$, then $\mathbf{x}^*$ is a Kuhn-Tucker-Karush point.

Proof: Satisfaction of the first of the Kuhn-Tucker-Karush condition follows from the optimization of the individual decision elements in Step 5. To establish that the second and third optimality conditions are satisfied, we must prove that values of the Lagrange multipliers corresponding to a constraint g_k are well defined, even though they may be determined by the solution to more than one subproblem. We establish this result by analyzing the convergence criterion, which requires

$$df/dx_i = \partial f/\partial x_i + \Sigma \lambda_k \partial g_k/\partial x_i = 0,$$

where x_i is a parameter. This equation is similar in form to the optimality condition

$$\partial f/\partial x_i + \Sigma \lambda_k \partial g_k/\partial x_i = 0,$$

except that x_i appears in the optimality condition as a design variable. Also, λ_k does not have the same meaning in both equations. We need to distinguish between the optimal values of the Lagrange multipliers λ_k for the entire problem and those for the local problem on which df/dx_i is based. Denote a λ_k for the entire problem by λ_k*. We will need to distinguish between locally determined λ_k 's and df/dx_i 's defined by different decision elements, so index these $\lambda_k{}^1$ and df^1/dx_i.

Let D_1 contain a design variable x_i, which appears explicitly in the constraint g_k. Define λ_k to be the value $\lambda_k{}^1$ determined from the optimal solution to D_1. We show that $\lambda_k{}^1$ defined in this way is unique, that is, the value of λ_k does not depend on the choice of a decision element containing g_k. We then show that $\lambda_k{}^* = \lambda_k$ exists satisfying the Kuhn-Tucker-Karush condition,

$$df/dx_i = \partial f/\partial x_i + \Sigma \lambda_k^* \partial g_k/\partial x_i = 0$$

for any i. This will establish the result.

Uniqueness of the λ_k 's follows from the requirement that the number of parameters passed from decision element D_1 to decision element D_2 is greater than or equal to the number of independent active constraints, which are in common to both decision elements. Let p_i, $i = 1,\ldots,n$, be parameters passed from D_1 to D_2. Then the convergence criterion specifies that the optimal sensitivity derivatives $df^2/dp_i = 0$, $i = 1,\ldots,n$. From the definition of the optimal sensitivity derivative we have

$$df^2/dp_i = \partial f/\partial p_i + \Sigma_{k \text{ in } D_2} \lambda_k^2 \partial g_k/\partial p_i = 0 \quad i = 1,\ldots,n.$$

The p_i are local design variables in D_1. Thus from the optimality conditions for D_1 we have

$$\partial f/\partial p_i + \Sigma_{k \text{ in } D_1} \lambda_k^1 \partial g_k/\partial p_i = 0 \qquad i = 1,\ldots,n.$$

Clearly, if g_k does not appear in D_1, then $\partial g_k/\partial p_i = 0$, since p_i is a design variable in D_1 and D_1 contains all of the constraints in which its design variables appear explicitly. Also, if g_k does not appear in D_2, we may arbitrarily set $\lambda_k^2 = \lambda_k^1$ without affecting the validity of the result. Making these changes and combining the two linear systems of n equations, we have

$$\Sigma_{k \text{ in } D1} (\lambda_k^1 - \lambda_k^2) \partial g_k/\partial p_i = 0 \qquad i = 1,\ldots,n$$

Since the constraints are independent, we have n equations in the m unknowns $\lambda_k^1 - \lambda_k^2$. By assumption, there are at least as many parameters, n, as there are independent constraints, m. Thus the linear system has either no solutions (overdetermined) or has a unique solution. The linear system has a solution if the decision element D_1 has a feasible, optimal solution. The system is homogeneous, so the unique solution is $\lambda_k^1 - \lambda_k^2 = 0$. Thus the λ_k's are well-defined. That $\lambda_k^* = \lambda_k$ satisfy the condition

$$\partial f/\partial x_i + \Sigma \lambda_k^* \partial g_k/\partial x_i = 0$$

follows from: (1) λ_k has the same value for all decision elements, and (2) the condition must be satisfied for a decision element containing x_i. This concludes the proof.

Proposition 2: Let the number of design variables in each decision element D_i equal or exceed the number of constraints in D_i which are independent and active at a point $\mathbf{x}^*$ determined by a feasible, optimizing decision sequence. Then if $\mathbf{x}^*$ is a Kuhn-Tucker-Karush point for the original problem P, all of the optimal sensitivity derivatives df/dx_i will be 0.

Proof. Certainly, for a decision element D_1 determining a design variable x_i, we have

$$\partial f/\partial x_i + \Sigma \lambda_k^1 \partial g_k/\partial x_i = 0\,,$$

as a consequence of the assumption that D_1 has been individually optimized in Step 5 of each iteration. D_1 must have at least as many design variables as independent active constraints, and from this we can conclude that $\lambda_k^1 = \lambda_k^*$, where the λ_k^* are the Lagrange multipliers corresponding to the Kuhn-Tucker-Karush point for the entire problem. We verify that $\lambda_k^1 = \lambda_k^*$ as follows: we have two linear systems

$$\partial f/\partial x_i + \Sigma \lambda_k^1 \partial g_k/\partial x_i = 0 \quad i = 1, \ldots, n_{D1}$$

$$\partial f/\partial x_i + \Sigma \lambda_k^* \partial g_k/\partial x_i = 0 \quad i = 1, \ldots, n_{D1}$$

where n_{D1} is the number of design variables in the decision element D_1. The first of these linear systems holds since D_1 is optimized during each solution pass through the sequence of design decisions. The second system holds as a consequence of the fact that $\mathbf{x}^*$ is a Kuhn-Tucker-Karush point. Since both of these systems have a unique solution (with the derivatives evaluated at $\mathbf{x}^*$), the system

$$\Sigma (\lambda_k^1 - \lambda_k^*) \partial g_k/\partial x_i = 0$$

has the unique solution $(\lambda_k^1 - \lambda_k^*) = 0$.

Consider the case where x_i is passed to a decision element D_2 as a parameter. We wish to show

$$df^2/dx_i = \partial f/\partial x_i + \Sigma \lambda_k{}^2 \partial g_k/\partial x_i = 0.$$

D_2 must also have at least as many design variables as independent active constraints, so we can reason as above that the Lagrange multipliers determined by the optimal solution of D_2, $\lambda_k{}^2$, are equal to the Lagrange multipliers corresponding to the Kuhn-Tucker-Karush point, $\lambda_k{}^*$. Then $\lambda_k{}^2 = \lambda_k{}^* = \lambda_k{}^1$.

Since we require (Step 4) that the number of parameters passed from D_1 to D_2 must equal or exceed the number of independent constraints in common to both D_1 and D_2, we know that solutions exist to the linear system:

$$\partial f/\partial x_i + \Sigma_{k \text{ in } D2} \lambda_k{}^2 \partial g_k/\partial x_i - \{\partial f/\partial x_i + \Sigma_{k \text{ in } D1} \lambda_k{}^1 \partial g_k/\partial x_i\} = df^2/dx_i .$$

Since $\partial g_k/\partial x_i = 0$ unless g_k appears in D_1, we can simplify this system to

$$\Sigma_{k \text{ in } D1} (\lambda_k{}^2 - \lambda_k{}^1) \partial g_k/\partial x_i = df^2/dx_i$$

$$\Sigma_{k \text{ in } D1} (\lambda_k{}^* - \lambda_k{}^*) \partial g_k/\partial x_i = df^2/dx_i$$

$$0 = \Sigma_{k \text{ in } D1} (0) \partial g_k/\partial x_i = df^2/dx_i .$$

Since our choice of decision elements was arbitrary, the result is established.

B. OPTIMAL SENSITIVITY DERIVATIVES AND PARETO-OPTIMALITY

We now turn to the justification for the equation relating optimal sensitivity derivatives and Pareto-optimality.

The Pareto-optimal solutions minimize

$$F = \Sigma\, \omega_i f_i , \quad \Sigma\, \omega_i = 1.$$

For the moment, fix the weights ω_i. Then the Lagrange multipliers are constant in the equations for the optimal sensitivity derivatives with respect to a parameter p ,

$$df_i/dp = \partial f_i/\partial p + \Sigma \lambda_j \partial g_j/\partial p .$$

Then, using $\Sigma\, \omega_i = 1$ and the linearity of the partial derivative,

$$\begin{aligned} dF/dp &= \partial F/\partial p + \Sigma \lambda_j \partial g_j/\partial p \\ &= \partial/\partial p\ \Sigma\, \omega_i f_i + \Sigma \lambda_j \partial g_j/\partial p \\ &= \Sigma\, \omega_i \partial f_i/\partial p + \Sigma\, \omega_i \Sigma \lambda_j \partial g_j/\partial p \end{aligned}$$

$$= \Sigma\, \omega_i\, [\partial f_i / \partial p \; + \; \Sigma\, \lambda_j \partial g_j / \partial p\,]$$

$$= \Sigma\, \omega_i\, df_i / dp \;\; ,$$

the result holding for an individual optimization problem, that is, without any decomposition. We call the expression dF/dp a Pareto-optimal sensitivity derivative.

Some care must be exercised in applying the equality

$$dF/dp = \Sigma\, \omega_i\, df_i / dp$$

across the subproblems of a decomposed problem. As in the rest of the theory developed in this appendix, the result requires the equality of corresponding Lagrange multipliers determined by separate optimization subproblems. We have only obtained the equality of the Lagrange multipliers across distinct subproblems when the derivatives

$$df_i / dp$$

(taken with respect to any local design variable, x_i, appearing in other decision elements as a parameter, p) are all 0, or alternatively from equality of the sensitivity derivatives at a Kuhn-Tucker-Karush point. In the developments to follow, we will be concerned with the case

$$dF/dp = 0,$$

so the result will hold across a decomposition for this objective function.

Finding a balancing sequence of design decisions is done at Step 3 of the meta-design procedure (see page B-4).

At this point, we have available the optimal sensitivity derivatives df_i / dp . The basic idea is to investigate the effect on the possible optimal decision sequences of changes in the priorities ω_i for the several design objectives. This naturally involves changing the ω_i's. Thus, we need to be concerned about the effect changes in the ω_i's may have on the values of the sensitivity derivatives.

1. Dependence of the Pareto-optimal Sensitivity Derivative on the Prioritization of the Multiple Objectives

Proposition 3: $dF/dp\,(\omega) = \Sigma\, \omega_i\, df_i / dp$.

Proof: We wish to know the effect of changing the prioritizations, ω_i, of multiple objective functions on the Pareto-optimal sensitivity derivative, dF/dp. We investigate the nature of the Pareto-optimal sensitivity derivative as a function of the ω_i's by computing the derivatives of $dF/dp\,(\omega)$. The ω_i's themselves are parameters, so we really are asking for the second optimal sensitivity derivative of F, with respect to ω_i and p, denoted here by

$$D_{p,\,\omega i}{}^{optimal}\,F\,.$$

That is, we wish to compute

$$dF/d\,\omega_i = \partial F/\partial\,\omega_i + \Sigma\,\lambda_j \partial g_j/\partial\,\omega_i \;=f_i$$

and then take the optimal sensitivity derivative of this quantity with respect to a parameter p. This second differentiation gives

$$D_{p,\,\omega i}{}^{optimal}\,F \quad = df_i\,/\,dp$$

Where $df_i\,/\,dp$ denotes the optimal sensitivity derivative of the objective function f_i with respect to the parameter p. All higher optimal sensitivity derivatives of F with respect to ω_i are zero.

Let the new values for the ω_i's be denoted by ω_{i1}. Then

$$\begin{aligned} dF/dp\,(\omega_1) &= \Sigma\,\omega_i\,df_i/dp \;+ \Sigma\,D_{p,\,\omega i}{}^{optimal}\,F \;\;\Delta\omega_i \\ &= \Sigma\,\omega_i\,df_i/dp \;+ \Sigma\,df_i/dp\;\Delta\omega_i \\ &= \Sigma\,(\omega_i+\Delta\omega_i)df_i/dp \\ &= \Sigma\;\omega_{i1}df_i/dp\,. \end{aligned}$$

The first equality holding since all derivatives higher than $D_{p,\,\omega i}{}^{optimal}F$ are zero, thus the first order Taylor series is exact. This is the desired result, since we have shown that the effect on the optimal sensitivity derivatives df/dp of changing the ω_i's can be computed by just formally changing their values in the expression $\Sigma\,\omega_i\,df_i/dp$.

2. Application to Methodology F

Propositions 1 and 2 are applied to methodology F of Chapter IV by noting that the optimality conditions $dF/dp_i\,(\omega) = 0$ for the approximate Pareto-optimization problem are the same as the convergence criteria for the solution of the exact Pareto-optimization problem by a parameter-passing scheme. We thus solve the exact Pareto-optimization problem when we satisfy these conditions. Proposition 3 allows us to satisfy the conditions by varying the relative prioritizations, while maintaining the significance of those conditions in terms of optimal sensitivity derivatives. Thus, propositions 1, 2, and 3, allow us to solve the exact problem by varying the relative prioritizations.

REFERENCES

B-1 Luenberger, D.G., *Linear and Nonlinear Programming*, Second Edition, Addison-Wesley, 1984.

B-2 Sobieszczanki-Sobieski, J.; *Optimization by Decomposition: A Step from Hierarchic to Non-Hierarchic Systems*, NASA TM-101494, September 1988.

APPENDIX C

CONTENTS

TABLES

FIGURES

LANDING GEAR LAYOUT FOR MINIMUM RETRACTION COMPLEXITY

This example illustrates the solution of an aircraft design problem using the procedure described in Appendix B. The role of approximation techniques in bringing numerical rankings of qualitatively different design alternatives into the design decision-making process is emphasized in this example. We compare the solution of the design problem using the method of Appendix B with a numerical optimization solution using the dual method of Schmit/Fleury. Once the solution is converged, the results are identical, as predicted by the theory developed in Appendix B.

Note that the optimal solution to this problem is found in a single pass through a properly planned sequential decision-making process.

A. AIRCRAFT SIZING AND INITIAL LAYOUT--MAIN LANDING GEAR LOCATION

In this examnle, we analyze the design problem of determining the location of the main landing gear (MLG) on a small, high-performance piston-engine racing aircraft. This design problem is extremely simple, yet elements of quantifying the qualitative are present. The problem is typical of engineering design problems in which constraints on the design are intended to preclude various failure modes.

There are several ways that an aircraft can fail when landing or on the ground. When the aircraft is taking off, landing, taxiing, or parked on the airfield, a gust can tip the aircraft over about a line from the MLG wheel location to the nose wheel location. To avoid this, the track angle (also called the lateral tip-over angle) is specified. Reference C-1 gives a value of 55 degrees as the maximum allowable for this angle. Track angle is related to the center of gravity (CG) location, track, wheelbase, MLG location, and nose landing gear (NLG) location by the equation:

$$\text{track angle} = \text{atan}[(\text{MLGz} - z_{CG})/((x_{CG} - \text{NLGx})*\sin(\text{atan}(\text{track}/(2*\text{wheelbase}))))]$$

Another failure mode can occur on landing. High-performance aircraft often approach the runway and touch down at relatively high angles of attack. If the center of gravity is behind the vertical plane of the main landing gear wheels, the moment of the weight of the aircraft, acting through the MLG wheels, about the center of gravity of the aircraft, will tend to sit the aircraft on its tail. This occurs as the weight of the aircraft is transferred from the wings to the landing gear. Restricting the angle between the center of gravity and the plane of the main landing gear wheels will cause the aircraft to hit its tail on the runway (and presumably bounce back) before it can come to rest in a stable position on its tail. This tail down angle (also called the longitudinal tip-over criterion) is related to the main landing gear and center of gravity locations by

$$\text{tail down angle} = \text{atan}((\text{MLGx} - x_{CG})/(\text{MLGz} - z_{CG}))$$

In any design project, the design team will have goals which reflect the competitive strategy the team seeks to execute in the design. In this example, such a goal is to minimize retraction complexity.

A basic principle of lightweight structural design is to keep the load paths simple. The loads encountered by the MLG are mostly absorbed by the shock strut, but part of the load will be transmitted to structural elements in either the wing or the fuselage. Using some of the the same structural elements to react both landing and in-flight loads results in a lighter structural weight. In-flight loads on the fuselage are primarily bending resulting from the inertia of the fuselage structure itself and the weight of the useful load carried in the fuselage as these masses are suspended from the wing structure; pressurization loads (if any); and torsional and bending loads resulting from aerodynamic forces on the horizontal and vertical control surfaces. Wing torsion and bending loads resulting from the aerodynamic forces on the wing and the inertia of the wing structure (and any fuel carried in the wing) are also transmitted to the fuselage.

In conventional semi-monocoque metal aircraft structural design, the wing and fuselage are stiffened shells formed by the aircraft skin, fuselage frames, longerons and bulkheads, wing spars, ribs, and other stiffeners. These elements are thin-walled and are designed basically to react shear and compressive end loads. The structural elements that are available to react the landing gear loads in this way are the ribs and spars in the wing and the longerons and frames in the fuselage. The load paths will usually be much simpler if the landing gear loads are applied to a fuselage frame or wing spar.

For many racing aircraft, a slight speed advantage can be gained by using as large a wing span as possible [Ref. C-2], subject to constraints on wing loading and static margin for acceptable handling qualities and on wing structural weight (which increases in rough proportion to the wing span). The wing structure will be lighter if the landing gear is attached to the fuselage. Thus the MLG location is constrained by the fuselage frame spacing. The fuselage frame spacing is constrained by producibility considerations (there must be adequate room to work between the frames) on the low end and by structural elastic stability on the high end.

The most important overall goal in the design of a piston-powered racer is to get as much horsepower as possible into an extremely low-drag package. These aircraft fly at relatively high dynamic pressures and tend to be lightweight to begin with, and as a result parasite drag may be as much as 125 times larger than induced drag, making aircraft wetted area, and hence aircraft size, much more critical than aircraft weight. The drag penalty for a fixed landing gear is not acceptable. Therefore the gear must be retractable and must fit into a small space. Reliability, maintainability, and cost will be adversely affected if complex retraction kinematics are required to accomplish this. Thus, minimizing retraction complexity is a goal for this example problem.

Retraction complexity is a qualitative rather than an inherently quantitative factor. In practice, a numerical ranking for such a factor would be assigned to each of several alternative designs by a team of landing gear, structural and aerodynamic designers, and producibility and supportability specialists. Methods for developing these rankings are outlined in reference C-3. For the purposes of this example, we assume a numerical ranking procedure has been used by a design team to assign rankings to four alternative landing gear (LG) arrangements as illustrated in Figure C-1.

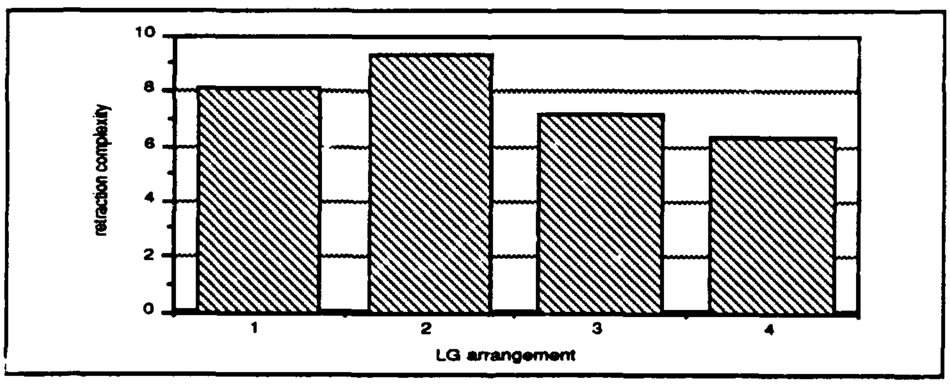


Figure C-1. Numerical Rankings of Retraction Complexity

The example problem becomes one of locating the main landing gear (in x, y, and z aircraft reference coordinates) in such a way as to minimize the retraction complexity, while satisfying constraints on tail down angle and track angle. The design variables are thus: MLGx, the x-coordinate of the main landing gear location; MLGy, the y-coordinate of the main landing gear location; and MLGz, the z-coordinate of the main landing gear location. Side constraints on each of the design variables MLGx, MLGy, and MLGz are derived as follows. The x coordinate of the aircraft CG location is at 9.5 ft. Thus a lower bound for MLGx is set at 10 ft. The upper bound is set at 15 ft. In a more detailed example, the upper bound would be related to the horizontal tail volume coefficient required to rotate the aircraft on takeoff. The lower bound for MLGy is set by the fuselage envelope at 1 ft. The upper bound for MLGy is related to the wing span and is set at 15 ft. The lower bound for MLGz is set by the shock strut length, which is related to the aircraft weight, design sink rate on approach, and landing gear design load factor, and is set at 4 ft. The upper bound for MLGz is related to maintenance access and is set at 7 ft. The statement of the example problem as an optimization problem is

Minimize: retractionComplexity (MLGx,MLGy,MLGz)

Subject to: trackAngle (MLGx,MLGy,MLGz) $\leq$ 55
tailDownAngle (MLGx,MLGy,MLGz) $\geq$ 15

$10 \leq \text{MLGx} \leq 15$

$1 \leq \text{MLGy} \leq 15$

$4 \leq \text{MLGz} \leq 7$

B. FORMULATION OF SUBPROBLEMS

The initial structure of design decision-making tasks is based on the idea that goals (such as minimize retraction complexity) and requirements (such as tail down angle $\geq$ 15 degrees) are propagated through choices of the design parameters MLGx, MLGy, and MLGz. In this example, each subproblem corresponds to the selection of a value for one of these design parameters. The relationships among attributes and constraints of the landing gear arrangement problem are represented by a directed graph in Figure C-2.

The initial subproblem structure is as shown in Figure C-3. Problem P1 (determination of MLGx) includes constraints that are linked in the directed graph of Figure C-2 by the design variable MLGx. Thus, since MLGx appears explicitly as a variable in both the tail down angle and track angle constraints, they are both included in problem P1.

MLGx does not appear explicitly in the retraction complexity objective, so it is not included in P1. Problem P2 contains constraints and the objective function in which the attribute MLGy appears explicitly as a variable. Problem P3 contains constraints and the objective function in which MLGz appears explicitly as a variable.

The meta-design problem is to formulate the subproblems P1, P2, and P3 to determine a convergent parameter-passing sequence for solving them and to identify an iteration strategy, if one is needed, to achieve optimality (or near-optimality). For this example, the decision support tool used to solve each of the problems is an x-y plot with the independent variable plotted on the x-axis and the objective function and constraints plotted on the y-axis. Other decision support tools, such as carpet plots, could be used to bring additional variables into each subproblem. Numerical optimization could also be used to solve the subproblems.

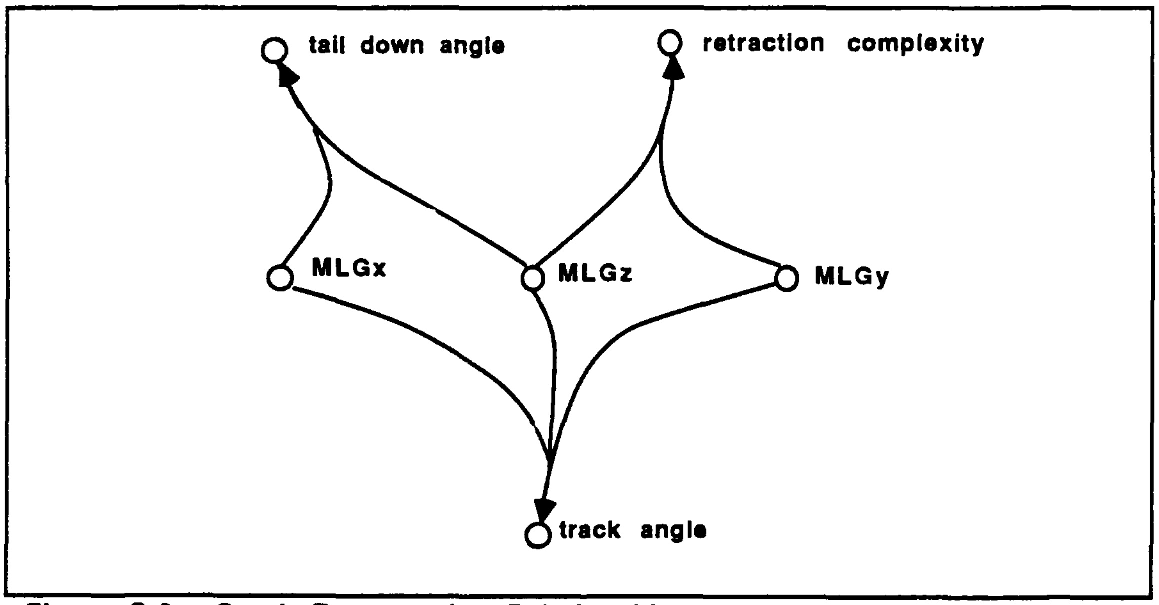
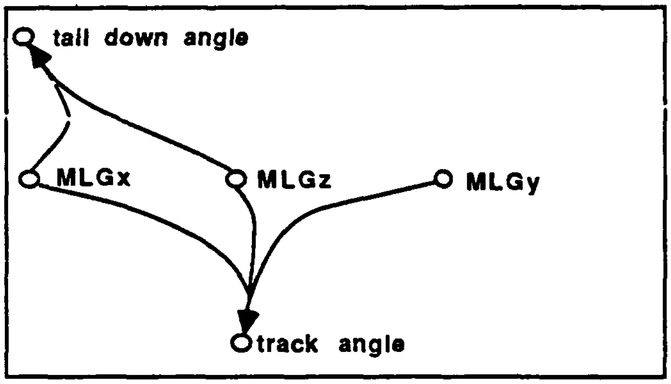
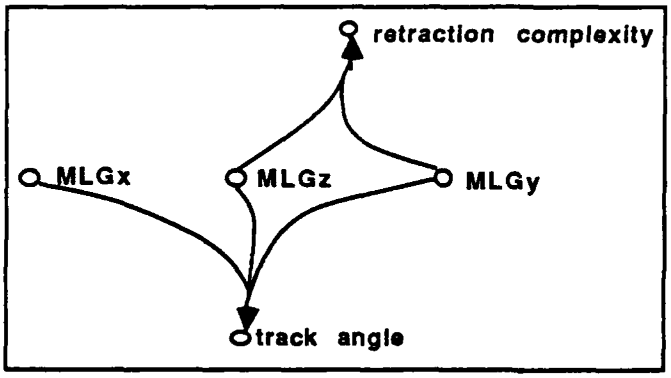


Figure C-2. Graph Representing Relationships Among Attributes for Landing Gear Arrangement Example



Problem 1

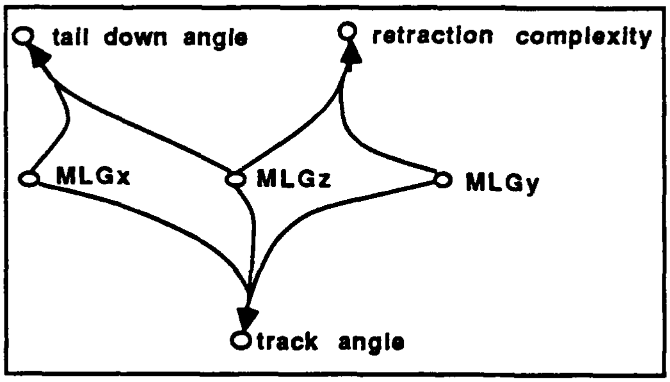
Satisfy:
track angle ≤ 55
tail down angle ≥ 15



Problem 2

Minimize:
retraction complexity

Subject to:
track angle ≤ 55



Problem 3

Minimize:
retraction complexity

Subject to:
track angle ≤ 55
tail down angle ≥ 15

Figure C-3. Initial Subproblem Structure

C. APPLICATION OF THE META-DESIGN PROCEDURE

The first step in the meta-design procedure is to solve the individual (one-dimensional search, in this case) optimization/constraint propagation problems P1, P2, and P3 as self-contained problems. This can be done using approximate forms for the goal and constraint design attributes.

1. Construction of Approximations

Simple mathematical expressions for track angle and tail down angle as a function of MLGx, MLGy, and MLGz were presented earlier in this chapter. In this section, these simple relationships are approximated by functions that are linear in either the design variables, x_i, or their inverses, $1/x_i$. This form of the approximating functions allows additional flexibility (relative to linear approximation) for approximating nonlinear design relationships, while retaining the highly advantageous property of convexity. The approximations are constructed by curve-fits to evaluation of the objectives and constraints on a set of alternative landing gear arrangements. This process allows rankings of the alternative designs based on the qualitative criterion of retraction complexity to be brought into the meta-design process on the same basis as the analytical models of track angle and tail down angle.

Many producibility and supportability considerations must be brought into the conceptual design process through numerical ranking of qualitative criteria. The fact that this approximation technique allows analytical performance and cost criteria to be balanced against producibility and supportability considerations is an ideal match to unified life cycle engineering (ULCE) and concurrent engineering objectives. Using the approximation technique, any aspect of the design that can be evaluated at some level can be brought into the trade-off process. This aspect of the approximation technique makes the approach highly suitable for use in aircraft preliminary design, detailed design, prototype testing, even production and initial support where closed form analytical solutions are often not accurate enough or available. Methods such as computational fluid dynamics, finite element analysis, wind tunnel tests, specimen tests, and, ultimately, operational evaluations would then provide the evaluation of alternative configuration, production, and support arrangements.

Results of evaluating tail down angle, track angle, and retraction complexity for each of four alternative landing gear arrangements are presented in Table C-1. MLGx, MLGy, and MLGz, the coordinates of the main landing gear location (specifically, the centroid of the main landing gear tire footprint), are used as independent variables and are also tabulated in Table C-1.

To maintain convexity, the coefficients of the design variables or their inverses must be positive for the objective function and for constraints in the form given in reference C-5. In this example, this criterion was used to determine whether a constraint or goal design attribute was directly or inversely proportional to a decision design attribute. Using this procedure, the following approximations were constructed:

$$\text{retraction complexity} \sim 0.6727*\text{MLGy} + 0.7751*\text{MLGz}$$

$$\text{track angle} \sim 55 - (1.795*\text{MLGx} + 78.91/\text{MLGy} + 4.554*\text{MLGz}) \geq 0$$

$$\text{tail down angle} \sim - 15 + 145.2 - (1190/\text{MLGx} + 3.916*\text{MLGz}) \geq 0$$

Table C-1. Results of Configuration Evaluations

Landing Gear Arrangement	1	2	3	4
MLGx (ft)	12.00	11.82	13.41	13.47
MLGy (ft)	6.27	6.87	3.88	3.88
MLGz (ft)	4.78	6.12	5.82	4.78
Tail down angle	27.00	21.00	34.00	40.00
Track angle	56.00	61.00	71.00	72.00
Retraction complexity	8.10	9.40	7.10	6.30

At least for this example problem, approximations can be found that provide a reasonably accurate qualitative picture of the design space. Example results for tail down angle (Figure C-4) are typical. These qualitative results are only valid locally, however, and can be quite inaccurate when extrapolated much beyond the region of design space near the alternative configurations that were evaluated to construct the approximations (Figure C-5).

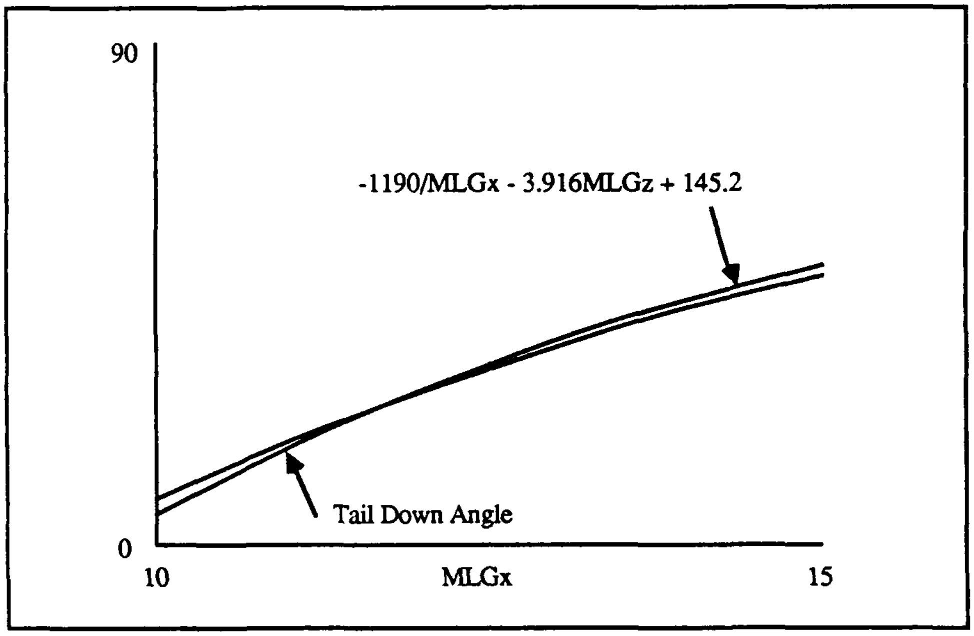


Figure C-4. Tail Down Angle as a Function of MLGx

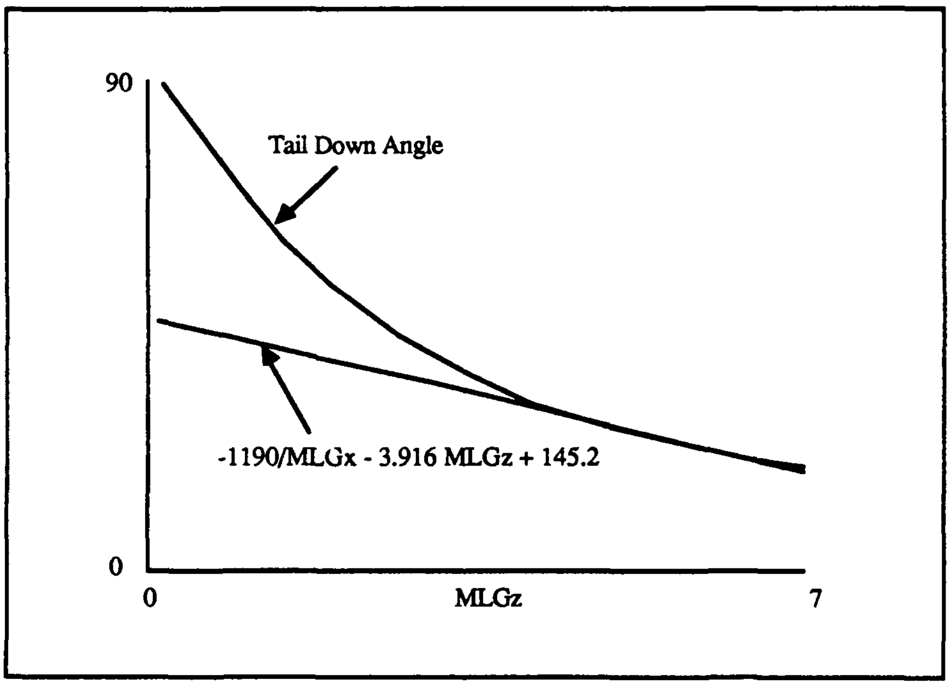


Figure C-5. The Approximations Are Not Usually Accurate Globally

2. Monotonicity Analysis

Problems P1, P2, and P3 are coupled by the objective function, retraction complexity, and by the tail down angle and track angle constraints. Thus it is important to know how the constraints and the objective function determine the design variables. This can be done using monotonicity analysis [Refs. C-6, C-7].

The monotonicity table for the landing gear layout problem is presented in Table C-2. For a design variable such as MLGx to be determined by the objective function and constraints, there must be both a "+" and a "-" in the MLGx column of the monotonicity table. A "+" is entered in the MLGx column at the track angle constraint row if the constraint is increasing with MLGx. (Note that the constraints must be written in a consistent way.)

Table C-2. Monotonicity Analysis for the Landing Gear Layout Example

	MLGx	MLGy	MLGz
Retraction complexity		+	+
15 - tail down angle ≤ 0	-		+
-55 + track angle ≤ 0	+	-	+
4 - MLGz ≤ 0			-

The following conclusions may be drawn from Table C-2:

- MLGx may be determined from the tail down angle and the track angle constraints.
- MLGy is determined by the track angle constraint and the objective function.
- MLGz is determined by its lower bound.

Using this information and the solutions to the one-dimensional subproblems, a convergent sequence for solving P1, P2, and P3 can be found. Initial solutions to P1, P2, and P3 using values from Landing Gear Arrangement 1 are shown in Figures C-6 through C-8.

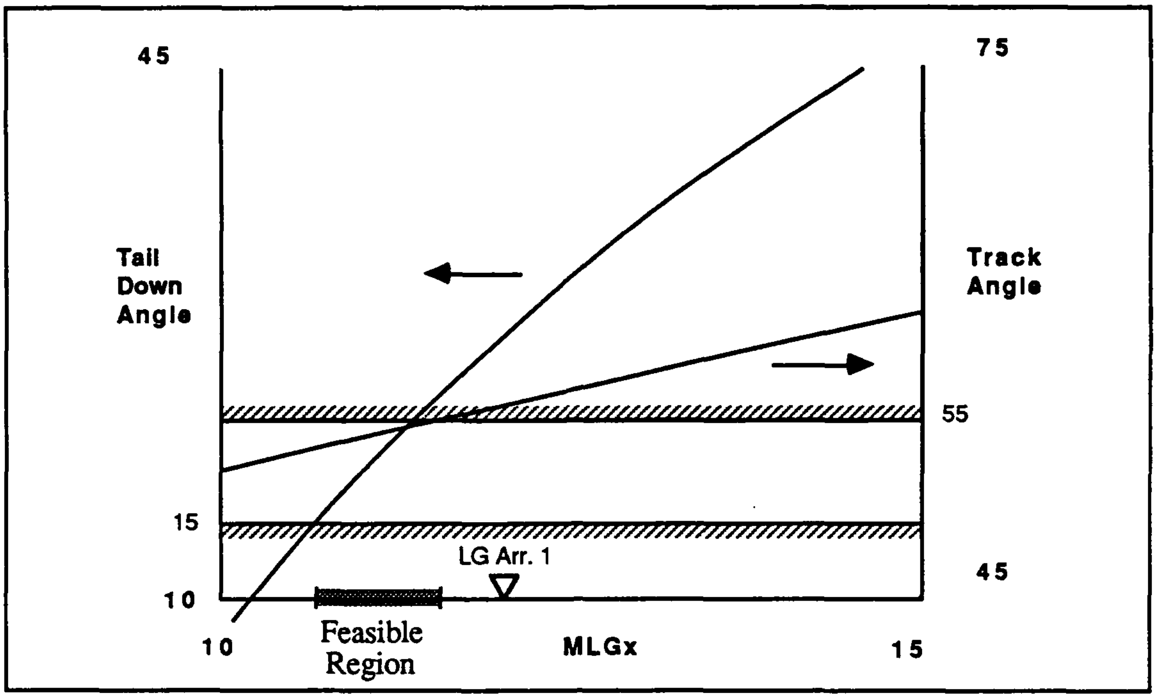


Figure C-6. Solution of Constrained Optimization Problem P1

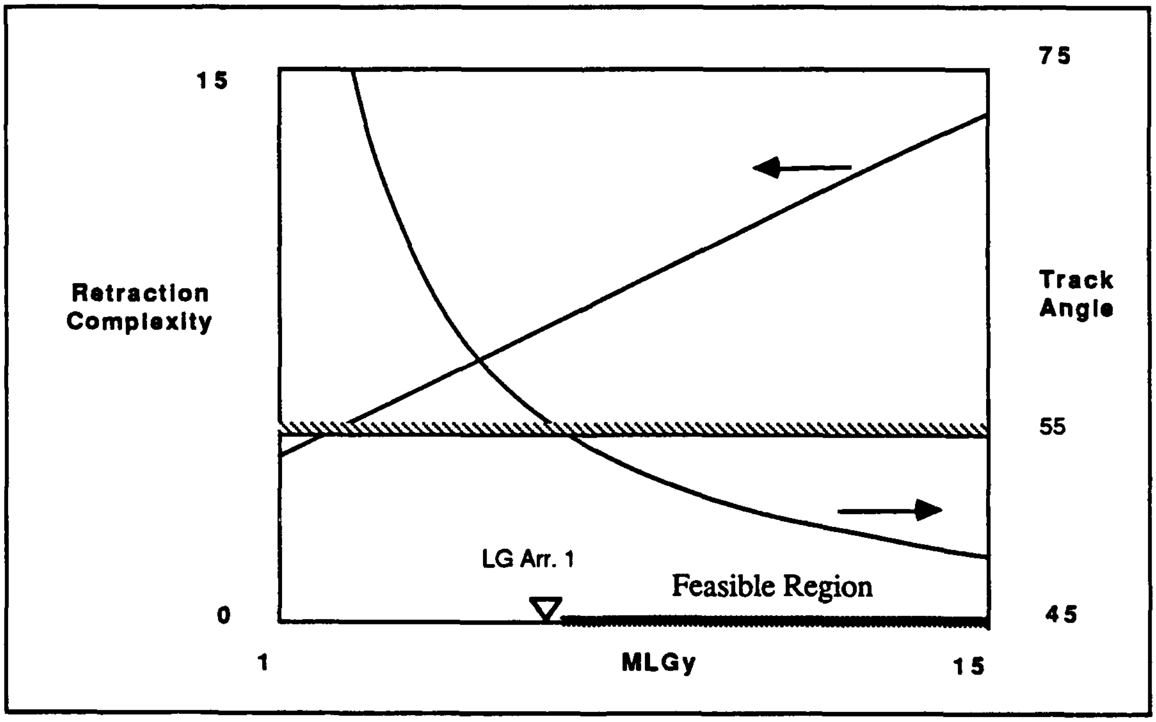


Figure C-7. Solution of Constrained Optimization Problem P2

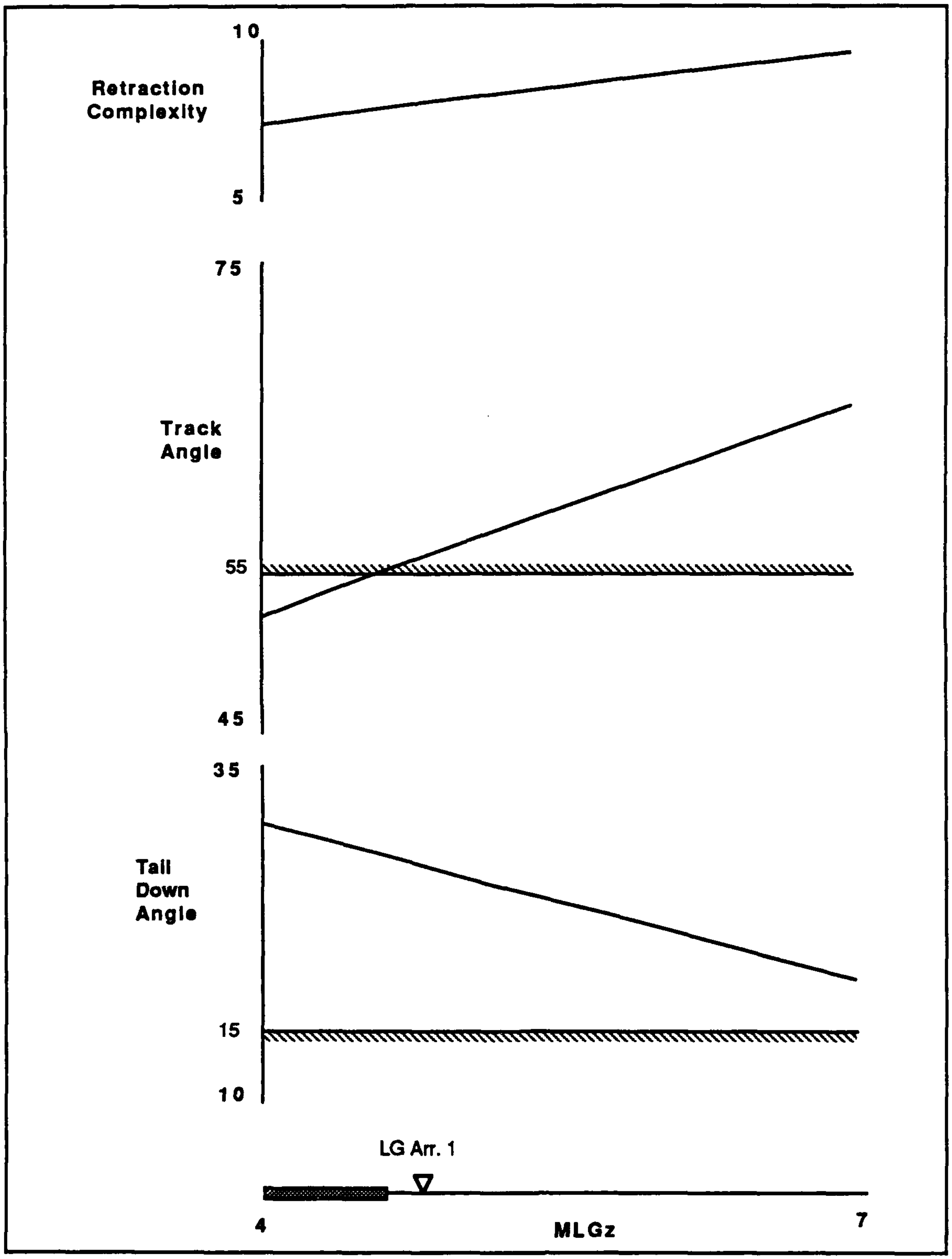


Figure C-8. Solution of Constrained Optimization Problem P3

Note that in Figure C-6 although monotonicity analysis indicated that MLGx may be determined by track angle and tail down angle, there is a range of values for MLGx that will satisfy both constraints.

The meta-design sequence for solving P1, P2, and P3 must first maintain feasibility. This means that if we specify that P1 is to be solved before P3, the value for MLGx selected in P1 must not result in a situation where there are no values of MLGz that will satisfy the constraints of P3.

To evaluate parameter passing schemes for feasibility, we examine the isolated solutions of Figures C-6 through C-8. For P1, we see that MLGx cannot be precisely determined from P1 in isolation, but that MLGx should be decreased from the Landing Gear Arrangement 1 value to find a feasible solution. Similarly, MLGy should be increased from 6.27 to 6.75, and MLGz should be decreased from 4.78 to 4.

The next step is to examine the effect that these changes in the design variables would have on the other subproblems. This can be done by taking partial derivatives of the constraints. A first order Taylor series approximation to the constraint then indicates whether a change in a design variable will make a constraint more or less critical. As an example, since ∂(tail down angle)/∂(MLGx) is positive, decreasing MLGx will decrease the tail down angle. Since tail down angle must be greater than 15 degrees, this change in MLGx makes the tail down angle constraint more critical for P3. Decreasing MLGz will make the tail down angle constraint less critical for P1. Thus P1 should not be solved before P3 if we wish to ensure feasibility.

The track angle constraint will actually be made less critical if the changes in the design variable values indicated by Figures C-6 through C-8 isolated subproblem solutions are made.

Once restrictions on the possible meta-design solution sequences associated with maintaining feasibility have been identified, consideration can be given to how the solution of the subproblems can be sequenced to lead to an optimal, or near-optimal, solution to the overall problem. This is done using the optimal sensitivity derivatives of the solutions to the one-dimensional subproblems.

To compute these, note that the objective function does not appear in P1, thus the optimal sensitivity derivatives for P1 are all 0. Also, ∂(retraction complexity)/∂(MLGx) = 0, and since none of the constraints (track angle or tail down angle) are active at the isolated optimal solution of P3 (Figure C-8), the optimal sensitivity derivative D^{opt}(retraction complexity)/D(MLGx) = 0 for passing MLGx to P3. (This case was already excluded because the results might be infeasible.) Also,

D^{opt}(retraction complexity)/D(MLGy) = ∂(retraction complexity)/∂(MLGy) for P3.

In the case of P2, the track angle constraint is active with Lagrange multiplier $(0.6727 * 6.75^2)/78.91 = 0.388$. Then

D^{opt}(retraction complexity)/D(MLGx) = 0.388*1.795 = 0.697

and

D^{opt}(retraction complexity)/D(MLGz) = (0.7751 + 0.388*4.554) = 2.54

These results are summarized in Figure C-9.

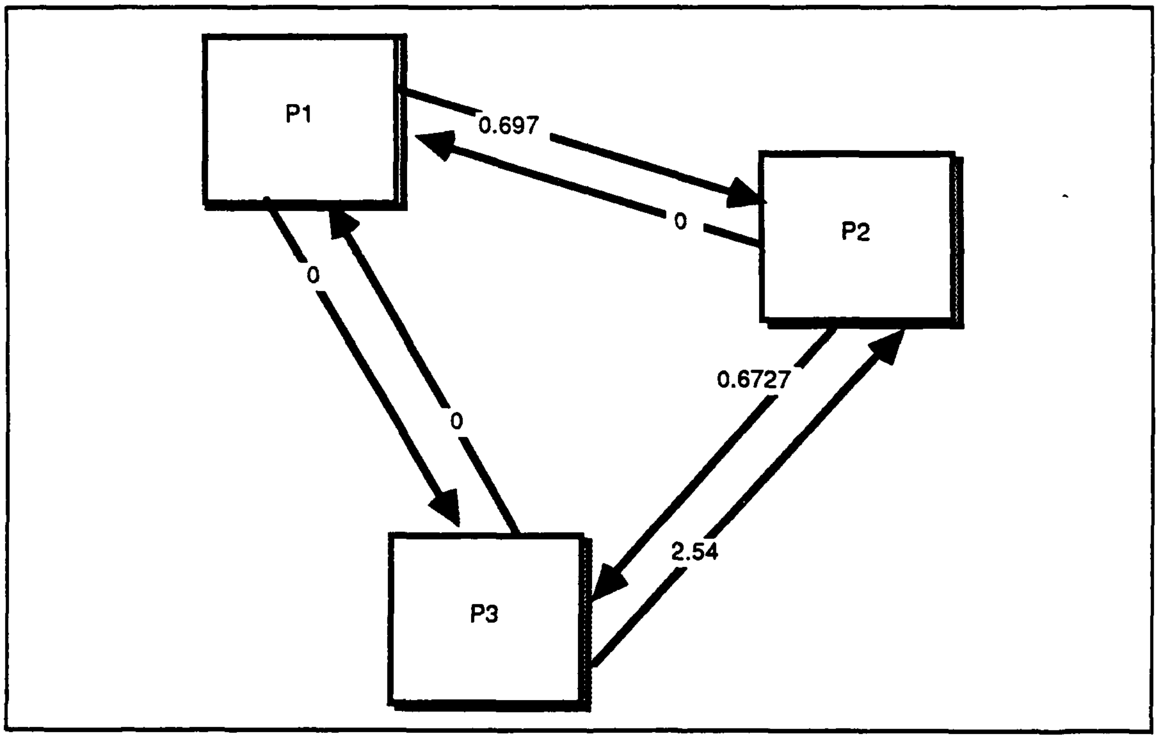


Figure C-9. Optimal Sensitivity Derivatives for Parameter Passing Schemes

To maintain feasibility, we have to solve P3 first, then P1. If we solve P2 before P3, the retraction complexity will actually increase by 0.6727*ΔMLGy. On the other hand, solving P3 before P2 will allow a reduction in the objective function by 2.54*ΔMLGy. Thus P3 should be solved before P2. Solving P2 before P1 will have no effect on the objective function, but solving P1 first, and then P2, will allow a reduction in the optimal value of the objective function that we can obtain in P2 by 0.697*DMLGy. Thus the best meta-design sequence will look like Figure C-10.

Note also that passing the new value for MLGy, obtained by solving P2, back through P1 and P3 will not improve the objective function. Thus, it is not possible to improve the objective function by iteration in this case.

Naturally, it is of considerable interest to see how closely the solution obtained using optimal constraint propagation with this meta-design sequence comes to the optimal

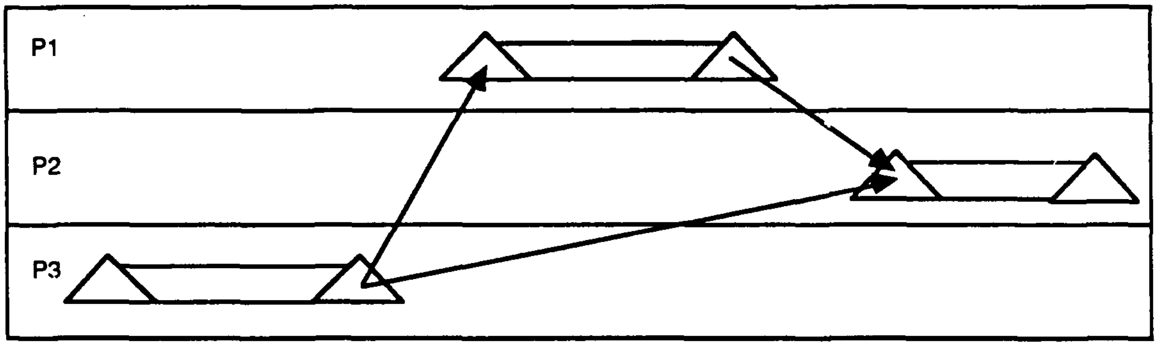


Figure C-10. Design Methodology for Landing Gear Layout Problem

solution. This comparison is made next, executing the meta-design sequence using optimal constraint propagation. This result is compared to an optimal solution obtained using the Schmit/Fleury technique.

3. Execution of the Meta-Design

The solution of the landing gear layout problem using the optimal constraint propagation proceeds as follows. First, P3 is solved as in Figure C-8, and MLGz is determined to be 4. This value is passed as a parameter from P3 to P1 and P2 as indicated by the sequencing arrows in Figure C-10. Following the meta-design plan, P1 is solved next, now with MLGz = 4.0. MLGy is still at the Landing Gear Arrangement 1 value of 6.27. However, the optimal value of the retraction complexity that can be obtained in a subsequent solution of P2 (as a function of MLGx) is used as a supplementary objective function for P1 (Figure C-11). This problem yields MLGx = 10.39. MLGx = 10.39 is then passed as a parameter to P2 (as indicated in the decision-making sequence shown in

Figure C-10) and P2 is now solved (Figure C-12), giving MLGy = 4.35 and a final value of the retraction complexity objective of 6.02.

Solution: MLGx = 10.39
MLGy = 4.35
MLGz = 4.0

retraction complexity = 6.02

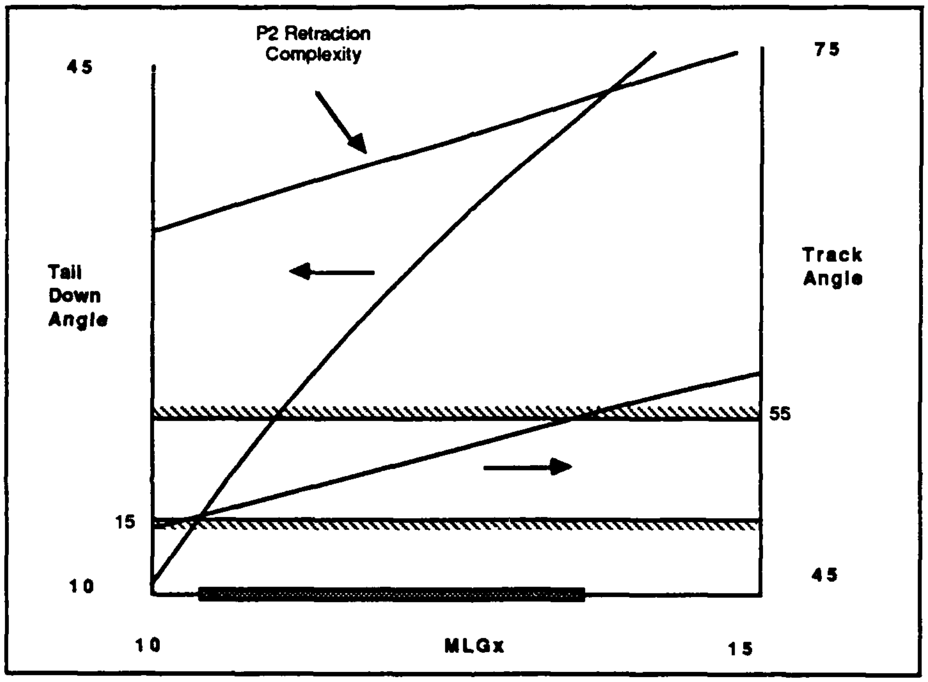


Figure C-11. Solution of P1, Constrained by Optimality of P2

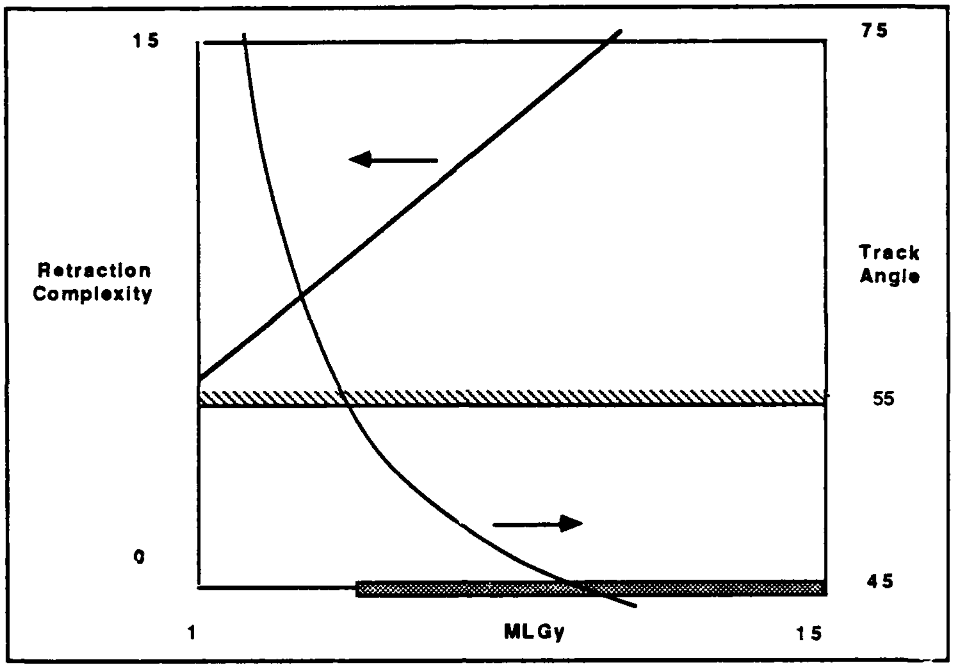


Figure C-12. Re-Solution of P2, Using Values for Parameters Determined in Previous Decision Elements

D. OPTIMAL SOLUTION USING DUAL METHODS AND APPROXIMATION TECHNIQUES

Dual methods can be applied to give meaning to the idea of computing optimal sensitivity derivatives when the parameters take on values in a discrete set. The details of accomplishing this are beyond the scope of the present work. However, this connection between optimal constraint propagation and dual methods makes it worthwhile to include some discussion of how approximate problems of the form used here can be quite easily solved using the technique of Schmit/Fleury. Since the optimal solution of the landing gear layout problem was needed for comparison with the optimal constraint propagation solution, solution of this problem using the Schmit/Fleury dual method is included here as an example.

Consider the solution of the approximate problem form considered in Ref. C-5:

The design variables are α_b.

The objective function (to be minimized) is approximated by

$$\Sigma_b w_b \alpha_b^{-1},$$

and the constraints by

$$b_q - \Sigma_b C_{bq}\, \alpha_b \geq 0.$$

The Lagrangian for this problem is:

$$\Sigma_b\, w_b\, \alpha_b^{-1} - \Sigma_q\, \lambda_q\, (b_q - \Sigma_b\, C_{bq}\, \alpha_b)$$

and the dual problem can be stated as:

$$\max_{\lambda}\, \min_{\alpha}\, \{\Sigma_b\, w_b\, \alpha_b^{-1} - \Sigma_q\, \lambda_q\, (b_q - \Sigma_b\, C_{bq}\, \alpha_b)\}$$

subject to:

$$\alpha^{min} \leq \alpha_b \leq \alpha^{max}$$

$$0 \leq \lambda_q.$$

(The particular form of the Lagrangian, $f - \Sigma\lambda g$ is a consequence of one of the Kuhn-Tucker-Karush optimality conditions for the primal problem.)

If a unique solution to the minimization over α, for a fixed value of λ, exists, solution of this minimization problem implicitly defines a function $\alpha(\lambda)$. For the explicit approximate problem form above, $\alpha(\lambda)$ is given as follows:

Solving

$$\frac{\partial}{\partial \alpha_b}\left\{\Sigma_b\, w_b^{-1} - \Sigma_q\, \lambda_q \left(b_q - \Sigma_b\, C_{bq}\, \alpha_b\right)\right\} = 0$$

gives

$$\alpha_b^2 = w_b/\{\Sigma\, \lambda_q\, C_{bq}\}.$$

Define

$$\alpha_b^2 = w_b/\{\Sigma\, \lambda_q\, C_{bq}\}.$$

If the minimization solution is at an interior point (i.e., $\alpha^{min} < \alpha_b < \alpha^{max}$ is satisfied as a strict inequality) then

$$\alpha_b^2 = a_b^2$$

Otherwise, if $a_b^2 > [\alpha^{max}]^2$, then $\alpha_b = \alpha^{max}$, and if $a_b^2 < [\alpha^{min}]^2$, then $\alpha_b = \alpha^{min}$. Defining $\alpha(\lambda)$ in this way allows us to write the dual objective function as:

$$\phi(\lambda) = \Sigma_b \, w_b \, \alpha_b(\lambda)^{-1} - \Sigma_q \, \lambda_q \, (b_q - \Sigma_b \, C_{bq} \, \alpha_b(\lambda))$$

Solution of the original problem can thus be reduced to solution of the simpler problem

max $\phi(\lambda)$

subject to: $\lambda \geq 0$

This basic approach can be used for other approximate forms for the objective and constraint functions. In particular, consider an objective function having the form $\Sigma_b w_b \alpha_b$, and constraint functions with the form:

$$\Sigma_i \, C_{iq} \, \alpha_i + \Sigma_j \, C_{jq} \, \alpha_i^{-1} \qquad \text{[Eqn C-1]}$$

where i indexes the variables to which the constraint is directly proportional and j indexes the variables to which the constraint is inversely proportional (in this form of the approximation functions, the variables appear directly or inversely, but not both).

For this form of the approximation functions, a zero of the derivative of equation C-1 at an interior point is given by:

$$\alpha_b^2 = a_b^2 = \Sigma \, \lambda_q \, C_{bq} / \{ w_b + \Sigma \, \lambda_r \, C_{br} \}.$$

Solutions at α^{max} and α^{min} are found as before.

The primal form of the main landing gear layout problem is

minimize: 0.6727*MLGy + 0.7751*MLGz

subject to:

55 - (1.795*MLGx + 78.91/MLGy + 4.554*MLGz) $\geq$ 0
130.2 - (1190/MLGx + 3.916*MLGz) $\geq$ 0.
10 $\leq$ MLGx $\leq$ 15
1 $\leq$ MLGy $\leq$ 15
4 $\leq$ MLGz $\leq$ 7

The dual problem is then

maximize:

0.6727*MLGy + 0.7751*MLGz + λ_1*((1.795*MLGx

$+ 78.91/MLGy + 4.554*MLGz) - 55)$

$+ \lambda_2*((1190/MLGx + 3.916*MLGz) - 130.2)$

subject to: $\lambda_i \geq 0, i = 1,2$

We have

$$\textbf{min}\ \{(0.7751 + \lambda_1*4.554 + \lambda_2*3.916)*MLGz\} \text{ occurs at } MLGz = 4, \quad 4 \leq MLGz \leq 10$$

for all feasible λ_i, i = 1,2. However, both MLGx and MLGy potentially involve a trade-off between the objective function and the constraints. MLGx is defined as a function of λ_1 and λ_2

$$MLGx = \begin{cases} 10 \text{ if } 10^2 \geq r^2 \\ r \text{ if } 10^2 \leq r^2 \leq 15^2 \\ 15 \text{ if } r^2 \geq 15^2 \end{cases} \qquad \text{where } r^2 = \frac{\lambda_2^* (1190)}{\lambda_1^* (1.795)}$$

MLGy is a function only of λ_1

$$MLGy = \begin{cases} 1 \text{ if } 1^2 \geq r^2 \\ r \text{ if } 1^2 \leq r^2 \leq 15^2 \\ 15 \text{ if } r^2 \geq 15^2 \end{cases} \qquad \text{where } r^2 = \frac{\lambda_1^*\ 78.91}{0.6727}$$

Making these substitutions, the dual objective function is (ignoring the constant terms coming from MLGz)

$$14.57* \sqrt{\lambda_1} - 36.78* \lambda_1 + 92.44* \sqrt{\lambda_1}\sqrt{\lambda_2} - 114.58* \lambda_2.$$

The maximum of this function occurs at $\lambda_1 = 0.1614$, $\lambda_2 = 0.0263$. Thus an optimal solution is

$$\lambda_1 = 0.1614,\ \lambda_2 = 0.0263$$

$$MLGx = 10.39,\ MLGy = 4.351,\ MLGz = 4$$

$$f^* = 0.6727*4.351 + 0.7751*4 = 6.0$$

This solution agrees precisely with the solution to the problem obtained using the sequential design process developed earlier in this appendix.

REFERENCES

C-1 Stinton, Darrol, *The Design of the Aeroplane*, Van Nostrand Reinhold, New York, 1983.

C-2 Hall, Stan, "High Aspect Ratio Wings for Formula One Racers", *Sport Aviation,* Experimental Aircraft Association, Sept. 1988, pgs. 33-38.

C-3 Cralley, W. E.; Owen, David; Kuenne, R., et al. (ULCE DSS Working Group), *Decision Support Requirements in a Unified Life Cycle Engineering (ULCE) Environment*, Volume I: An Evaluation of Potential Research Directions, Institute for Defense Analyses, IDA Paper P-2064, May 1988.

C-4 Lockheed Missiles and Space Company, *System Engineering Management Guide*, Defense Systems Management College, Ft. Belvoir, VA. Contract MDA 903-82-C-0339, October 1983.

C-5 Fleury, C.; and Schmit, L. A., *Dual Methods and Approximation Concepts in Structural Synthesis*, NASA CR-3226, 1980.

C-6 Wilde, D.J., *Globally Optimal Design*, Wiley Interscience, New York, 1978.

C-7 Michelena, Nestor F., and Agogino, Alice M., *Multiobjective Hydraulic Cylinder Design*, Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 110, March 1988, pgs. 81-87.

APPENDIX D

CONTENTS

FIGURES

BALANCING COST, RISK, AND PERFORMANCE IN AIRCRAFT SIZING

In this appendix we solve a multiobjective aircraft conceptual design problem explicitly and compare this solution with one obtained using the theory developed in Appendix B. The theory developed in Appendix B allows us to determine the entire family of Pareto-optimal solutions using only the solution of a single optimization problem for each of the multiple objectives and partial derivatives. The explicit solution technique requires us to solve a new optimization problem to determine each Pareto-optimal solution in the family. The method used here is much more efficient than the explicit technique.

1. Pareto-Optimality

Pareto-optimal solutions for multiobjective optimization problems are solutions for which no one objective can be improved without making one of the other objectives less optimal. The selection of a Pareto-optimal solution then corresponds to a prioritization of the objectives (prioritization may include ranking all of the objectives equally). Pareto-optimality is thus an appropriate way of precisely stating the goal of balanced design.

The main practical difficulty in applying Pareto-optimality in developing a balanced design is in the expense of finding enough of the possible Pareto-optimal solutions. An approach to overcoming this limitation through design process planning is described here.

To solve a Pareto optimization problem (Ref. D-1) ,a new objective function, f, is defined by forming the product of each of the multiple objective functions, f_i , with a weighting factor for that objective function, ω_i, and summing over the index variable, i , for the multiple objective functions:

$$f = \Sigma\, \omega_i f_i$$

The minima of f are Pareto optimal solutions. We approach the problem of balanced design by applying the techniques of Appendix B to the objective function f. The tools we have available to design the design process are alternative groupings of design attributes into design decision elements and alternative design decision sequences. The results of Appendix B allow us to use those tools to develop a design process corresponding to a given prioritization of the multiple design objectives.

2. Problem Statement

This example conceptual design/sizing problem can be stated and solved explicitly as a multiobjective optimization problem:

minimize: $F(\omega, L/D, W_{TO}, BSFC)$

subject to:

$$\text{available\_fuel\_constraint}\ (W_{fuel}, W_{TO}) \geq 0$$

$$\text{required\_fuel\_constraint}\ (W_{fuel}, W_{TO}, BSFC, L/D, \eta_p, R) \geq 0$$

$$0 \leq L/D \leq 25$$

$$4000 \leq W_{TO} \leq 46{,}000 \text{ (lb)}$$

$$0.35 \leq BSFC \leq 0.95$$

$$2000 \leq W_{fuel} \leq 23{,}000 \text{ (lb)}$$

where

L/D	=	lift-to-drag ratio
W_{TO}	=	take-off weight
BSFC	=	brake-Hp specific fuel consumption
W_{fuel}	=	fuel weight
η_p	=	propeller efficiency
R	=	range.

For this example solution, η_p and R are taken to be problem parameters with fixed values:

$$\eta_p = 0.85$$

$$R = 25{,}000.$$

The available fuel constraint is based on a curve fit of fuel-burning long-range aircraft:

$$W_{TO} - W_{fuel} - 1.222\, W_{TO}^{0.8382} \geq 0$$

The required fuel constraint is derived from the Breguet range equation:

$$W_{fuel} - W_{TO}\left(1 - e^{-[R\, BSFC/(375\, \eta_p\, L/D)]}\right) \geq 0$$

The objective function, F, is the weighted sum of three conflicting goals:

minimize W_{TO}

L/D as low as possible (below 25)

BSFC as high as possible (above 0.35).

One way to formulate F to balance these goals is to set

$$F = \omega_1/(1 - (L/D/25)) + \omega_2/((BSFC/0.35) - 1) + (1 - \omega_1 - \omega_2)\, W_{TO}/10000$$

Minimization of F with the value of ω fixed will allow us to find Pareto optimal solutions.

a. Discussion of the Problem Formulation

The maximum lift-to-drag ratio for a subsonic aircraft is given by

$$L/D_{max} = (b/2)*\sqrt{(\pi e/f)}$$

where b is the wing span, e is Oswald's planform efficiency factor, and f is the parasite (or equivalent flat plate) area.

Achieving a high L/D_{max} requires a combination of advanced aerodynamic and structural design. The maximum bending stress in the wing structure is directly proportional to the wing span, so increasing the wing span will generally require either advanced materials (increasing development risk and cost) or additional structural weight.

The planform efficiency may be increased by tailoring the planform shape, airfoil section, and geometric twist to optimize the span loading. Each of these developments makes the aircraft more difficult to manufacture. The wing can also be expected to twist under torsional loads encountered in cruising flight, and since the magnitude of these loads changes with differing altitudes and cruise speeds, matching the aerodynamic and structural characteristics of the wing is a complex multidisciplinary optimization problem.

The parasite area (zero-lift drag coefficient multiplied by reference area; see, for example, reference D-2) can be reduced by making the aircraft longer or increasing cruise speed, (thus increasing cruise Reynolds number) as long as the wetted area is not increased and the transonic drag rise is avoided.

The conclusion is that although it is desirable to make the lift-to-drag ratio as high as possible from a performance point of view, manufacturing and engineering development cost and risk considerations drive us to keep L/D as far below the upper limit of 25 as possible.

In a similar argument, considerations of engine life and propulsion system development cost and risk drive us toward specification of a brake-horsepower specific fuel consumption as far above the target value of 0.35 as possible. The Pareto-optimization problem is thus to balance superior performance (as quantified by minimum take-off gross weight to perform the mission) against development cost and risk and operational life.

b. Explicit Solution

Fuel required and fuel available are graphed as a function of take-off gross weight (W_{TO}) in Figure D-1. Feasible solutions are to the right of the dashed line on the take-off gross weight axis and lie between the fuel required and fuel available curves. Since these curves are typical, the minimum W_{TO} solution for any values of L/D and BSFC will lie at the intersection of these curves. Thus, both constraints are active.

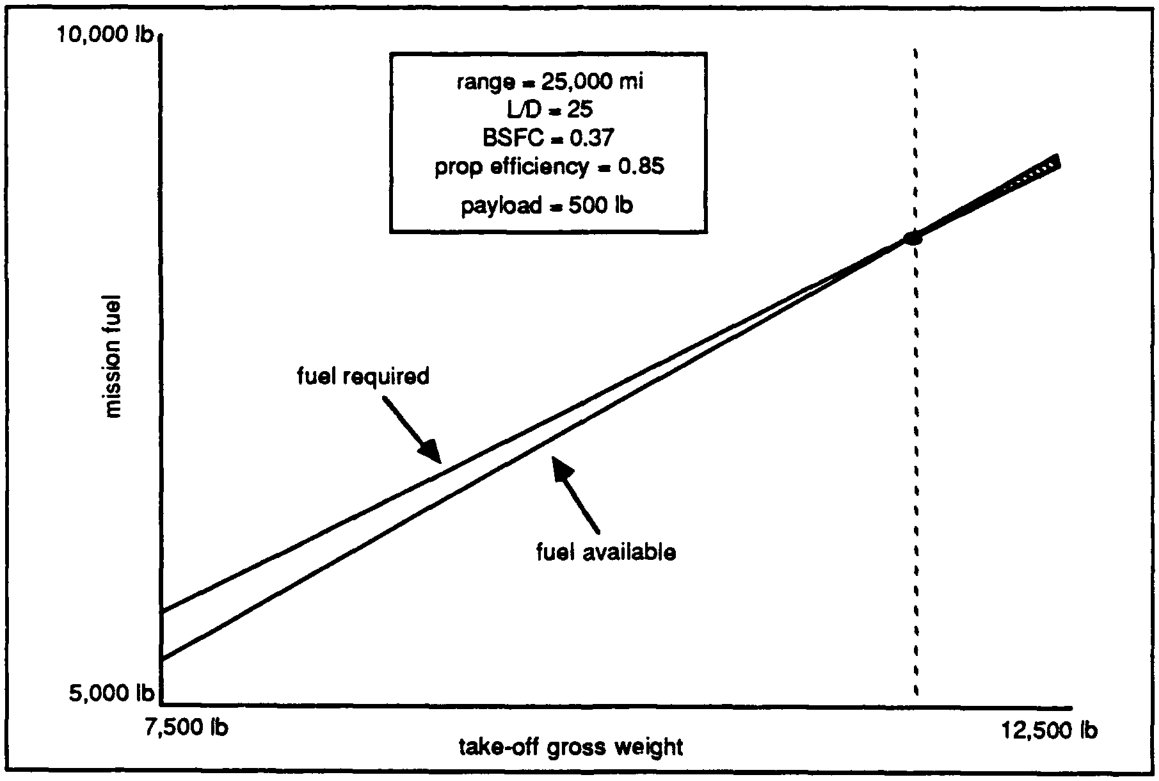


Figure D-1. Fuel Required/Fuel Available Problem.

With both constraints active, they are both equal to zero and, hence, equal to one another. This allows us to obtain a relationship for BSFC in terms of L/D and W_{TO}:

$$BSFC = 60.675\, \eta_p\, L/D\, (\ln W_{TO} - 1.2372)/R.$$

Using this relationship, we can find interior extrema of the objective function $F(\omega, L/D, W_{TO}, BSFC)$ by setting the partial derivatives $\partial F/\partial L/D$ and $\partial F/\partial W_{TO}$ equal to zero. To compare solutions obtained using this approach to those using the method outlined in Chapter IV, we interpret these optimality conditions as equations to determine the values of the relative prioritizations ω_i corresponding to various values of L/D, W_{TO}, and BSFC. The difference between the approach we are taking here, and the approach of Chapter IV, is that here we have found the solution to the Pareto optimization problem explicitly. This may involve considerable computation for more complex problems.

We represent the entire family of Pareto-optimal solutions by solving the equations for ω_1 and ω_2. Then we have two equations in the two unknowns ω_1 and ω_2:

$$\mathbf{A}\ \omega = \mathbf{b}$$

where

$$a_{11} = 1/(25(1 - L/D/25)^2)$$

$$a_{12} = -BSFC/(0.35\ L/D\ (BSFC/0.35 - 1)^2)$$

$$a_{21} = -1/10{,}000$$

$$a_{22} = [-60.675\ \eta_p\ L/D/(0.35\ R\ W_{TO}\ (BSFC/0.35 - 1)^2)] - 1/10000$$

and

$$b_1 = 0,\ b_2 = -\ 1/10000.$$

These solutions can be presented by plotting the values of ω_1 and ω_2 as L/D varies for fixed values of W_{TO} (Figure D-2) or as W_{TO} varies for fixed values of L/D (Figure D-3) or BSFC (Figure D-4) (using the relationship for BSFC in terms of L/D and W_{TO}). Pareto optimal solutions are located where the three curves intersect. For example, in Figure D-5, W_{TO} = 20,000 lb, BSFC = 0.4, and L/D = 22.4 is a Pareto optimal solution with $\omega_1 = 0.14$, $\omega_2 = 0.20$, and $\omega_3 = 1 - \omega_1 - \omega_2 = 0.66$.

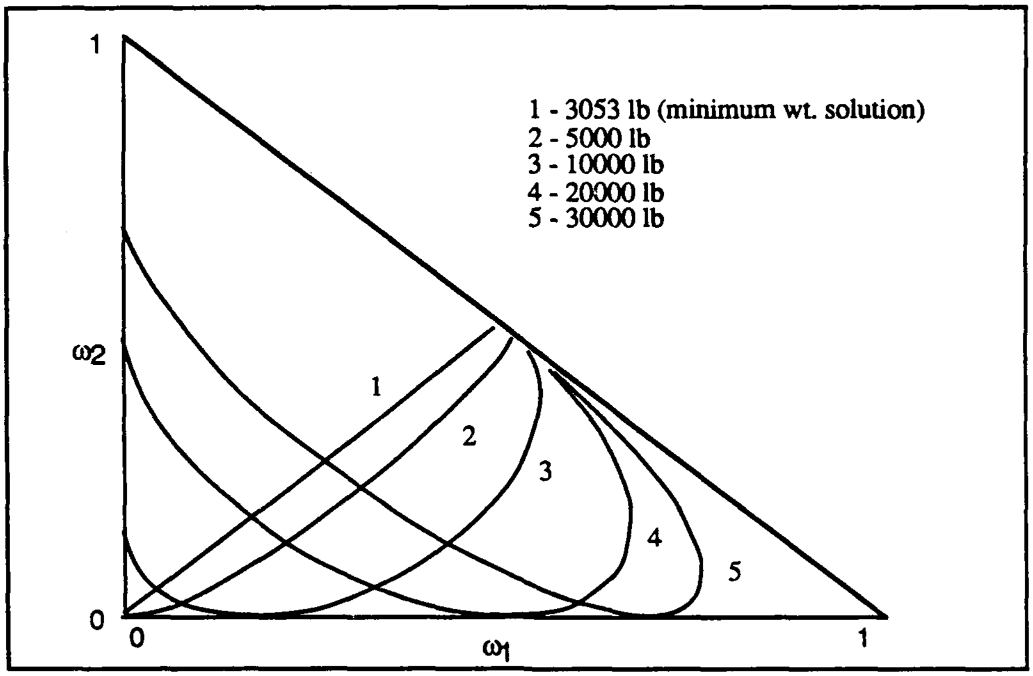


Figure D-2. Pareto-Optimal Solution Curves - Constant Take-Off Gross Weight

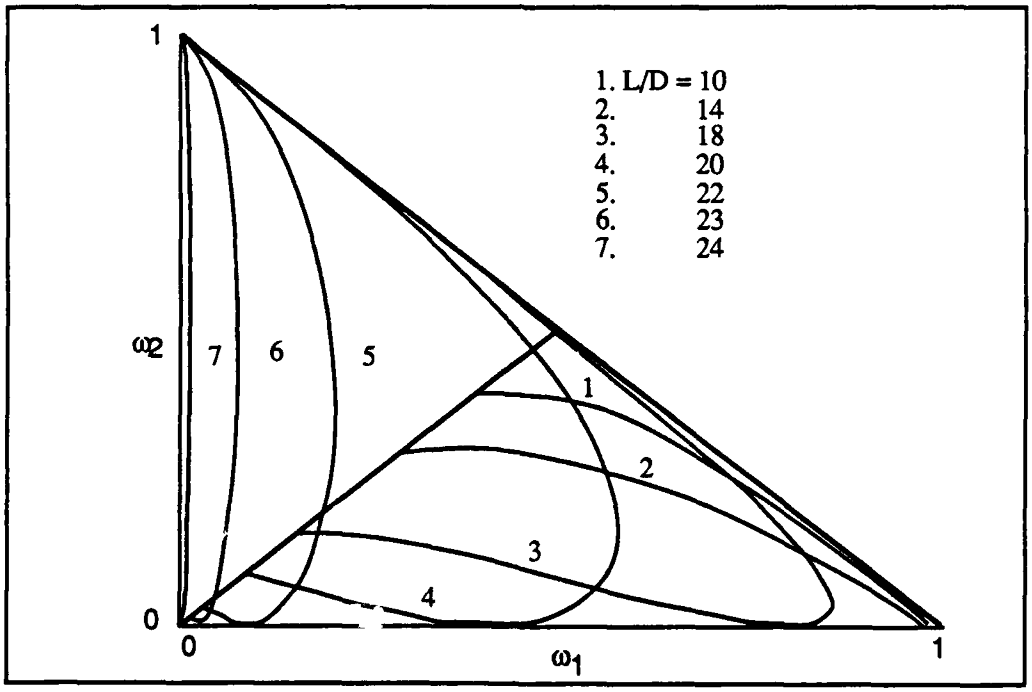


Figure D-3. Pareto-Optimal Solution Curves - Constant L/D

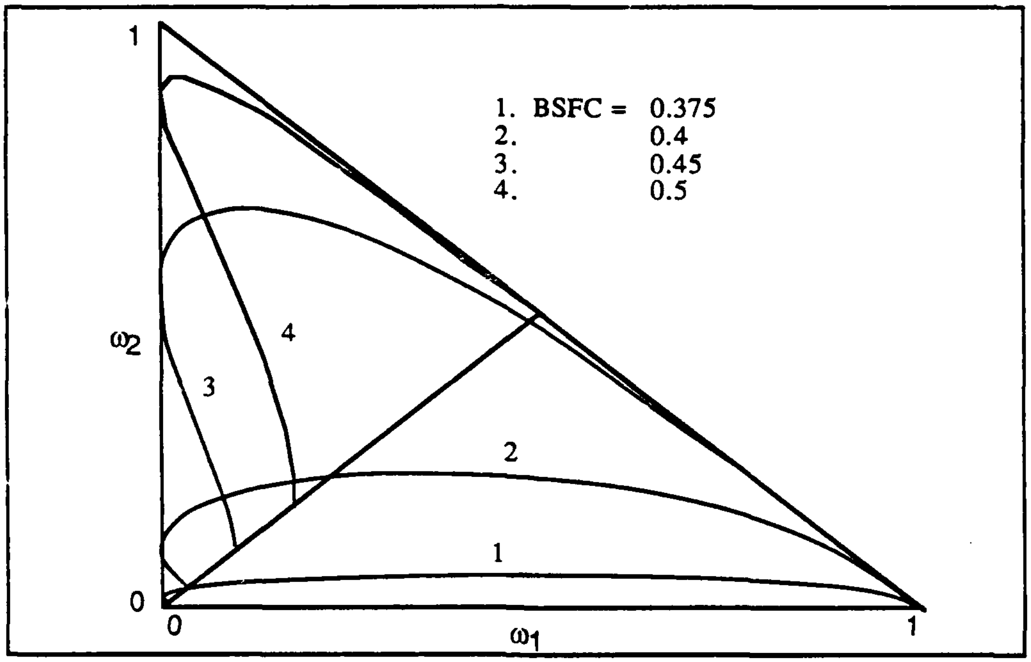


Figure D-4. Pareto-Optimal Solution Curves - Constant BSFC

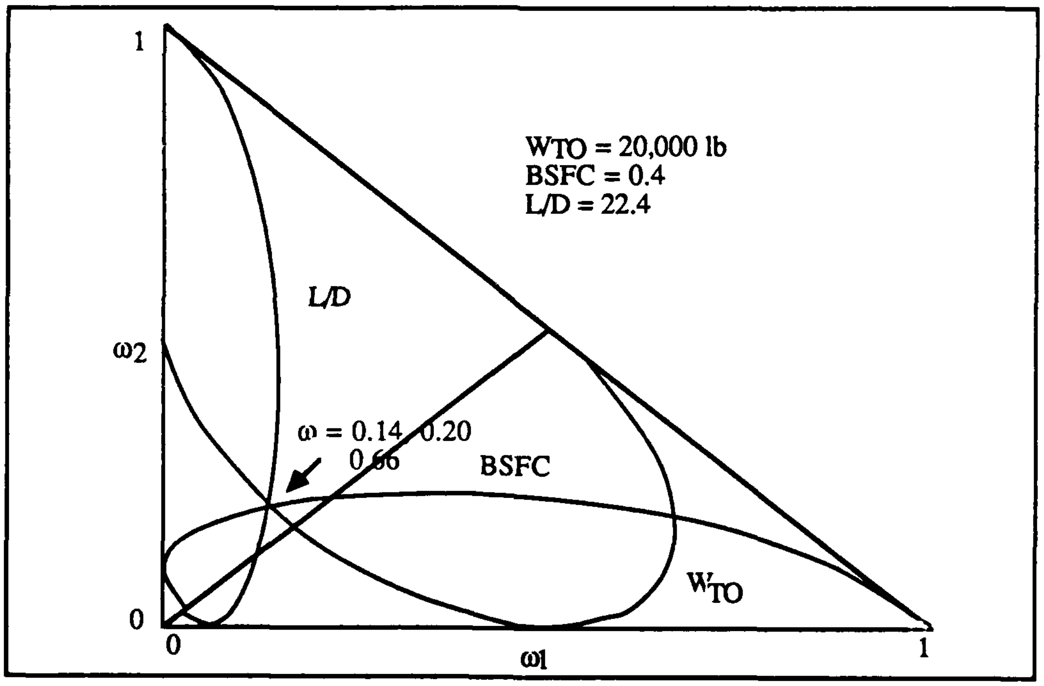


Figure D-5. A Pareto-Optimal Solution

3. Optimal Sensitivity Derivatives

Optimal sensitivity derivatives can be used to integrate several coupled design decision-making tasks that are performed in sequence. The situation considered in the development of the optimal sensitivity derivatives is shown schematically in Figure D-6. Three interrelated design decisions must be integrated: choosing L/D (lift-to-drag-ratio), choosing BSFC, and choosing W_{fuel} and W_{TO}. The decisions are coupled by the fact that choices for W_{fuel} and W_{TO} are restricted by the fuel required constraint, which in turn depends on the values of L/D and BSFC.

The optimal sensitivity derivative technique is based on the idea that each design variable is fixed by a specific decision. The design variable appears as a parameter in subsequent decisions. The distinction between a parameter and a local design variable is essential. For example, the lift-to-drag ratio is fixed by the "choose L/D" decision [represented as a plot of the lift-to-drag ratio penalty function (1/(1 - (L/D/25))) vs. lift-to-drag ratio in the upper left-hand corner of Figure D-6]. Lift-to-drag ratio is a local design variable in making this decision. The lift-to-drag ratio subsequently appears as a parameter in the "choose W_{fuel} and W_{TO}" decision (plot of fuel required and fuel available constraints as a function of W_{TO} in Figure D-6). Parameter passing is thus a way to relate coupled decision tasks to one another.

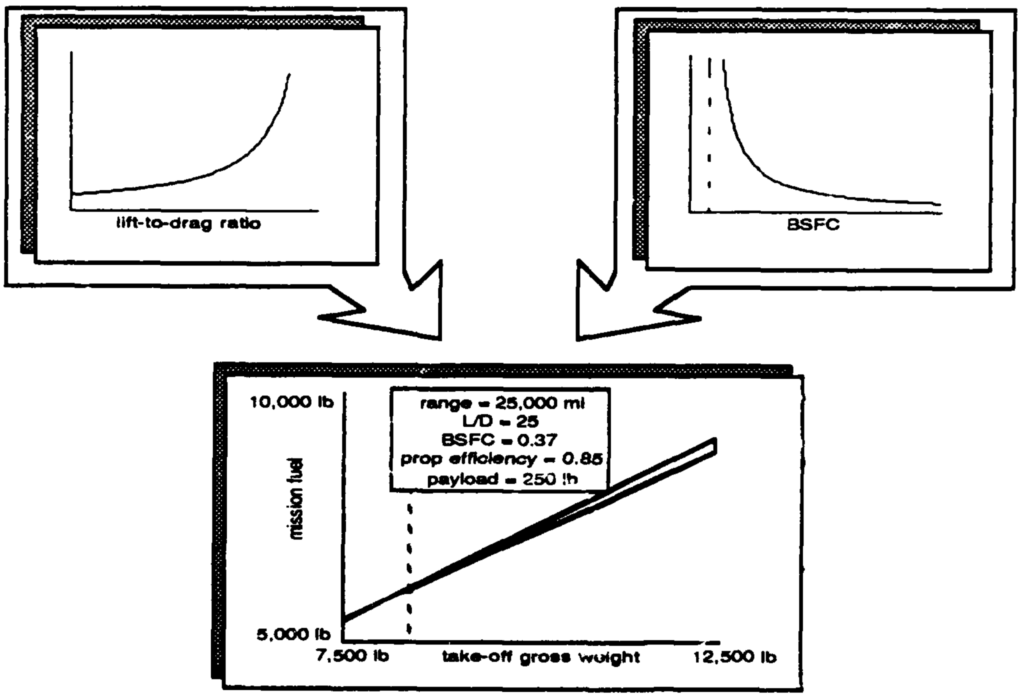


Figure D-6. Parameter Passing for Optimal Sensitivity Derivatives

It is important to know how optimal values of local objective functions change as the parameters are varied. For example, the minimum weight aircraft is found at the intersection of the fuel available and fuel required curves for the choose W_{fuel} and W_{TO} decision. This intersection point will shift up or down the fuel available curve as lift-to-drag ratio and brake-Hp specific fuel consumption are varied. Quantifying these rates of change is the purpose of the optimal sensitivity derivatives. The optimal sensitivity derivatives thus allow us to define the (constrained) optimal value of an objective function, such as W_{TO}, as a function of a parameter such as lift-to-drag ratio.

Optimal sensitivity derivatives are computed by applying one of the Kuhn-Tucker-Karush necessary conditions for optimality and using the fact that the constraints remain satisfied to simplify the total derivative of the objective function with respect to the parameter. Thus, for an optimization problem

$$\text{minimize: } f(\mathbf{x},p)$$

$$\text{subject to: } \mathbf{g}(\mathbf{x},p) \geq \mathbf{0}$$

the total derivative of f with respect to p is

$$df/dp = \partial f/\partial p + (\partial f/\partial \mathbf{x})(d\mathbf{x}/dp),$$

recognizing that as p is varied, the values of the design variables **x** will be adjusted to maintain optimality and feasibility. The idea of the simplification is to avoid having to compute (dx/dp). Since the x's are adjusted to maintain feasibility, we have

$$d\mathbf{g}/dp = \partial \mathbf{g}/\partial p + (\partial \mathbf{g}/\partial \mathbf{x})(d\mathbf{x}/dp) = \mathbf{0},$$

so

$$\partial \mathbf{g}/\partial p = -(\partial \mathbf{g}/\partial \mathbf{x})(d\mathbf{x}/dp).$$

The necessary condition

$$\partial L(\mathbf{x},p,\lambda)/\partial \mathbf{x} = 0,$$

where $L(\mathbf{x},p,\lambda)$ is the Lagrangian, $f(\mathbf{x},p) - \lambda^T \mathbf{g}(\mathbf{x},p)$ relates

$$(\partial f/\partial \mathbf{x}) = \lambda^T(\partial \mathbf{g}/\partial \mathbf{x}).$$

Substituting, we have

$$df/dp = \partial f/\partial p + (\partial f/\partial \mathbf{x})(d\mathbf{x}/dp) = \partial f/\partial p + \lambda^T(\partial \mathbf{g}/\partial \mathbf{x})\,(d\mathbf{x}/dp) = \partial f/\partial p - \lambda^T\,\partial \mathbf{g}/\partial p.$$

The advantage of this technique is that we can compute the total derivative of f, maintaining optimality and feasibility, from partial derivatives and the Lagrange multipliers

λ. The Lagrange multipliers can be computed from the necessary conditions, so this procedure is usually more efficient than computing dx/dp directly, using reoptimization and finite differencing.

Optimal sensitivity derivatives can be computed for the sizing example. Consider the choose W_{TO} and W_{fuel} design decision-making task described above. The Lagrangian for this subproblem is

$$L(\mathbf{x},\mathbf{p},\lambda) = W_{TO} - \lambda_1(W_{fuel} - W_{TO}(1 - e^{-[R\ BSFC/(375\ \eta_p\ L/D)]}))$$
$$- \lambda_2 (W_{TO} - W_{fuel} - 1.222\ W_{TO}^{0.8382}).$$

Here, $\mathbf{x} = (W_{TO}, W_{fuel})$, $\mathbf{p} = (L/D, BSFC, R, \eta_p)$ (p is now a vector), and

$$\lambda = (\lambda_1, \lambda_2),\ \mathbf{g}(\mathbf{x}) = (W_{fuel} - W_{TO}(1 - e^{-[R\ BSFC/(375\ \eta_p\ L/D)]}),$$
$$W_{TO} - W_{fuel} - 1.222\ W_{TO}^{0.8382})$$

The optimal solution is given by

$$W_{fuel} = W_{TO} - 1.222\ W_{TO}^{0.8382}$$
$$W_{TO} = e^{[(R\ BSFC/(60.675\ \eta_p\ L/D)) + 1.2372]})$$

The Lagrange multipliers are found by solving:

$$\partial L/\partial W_{TO} = 1 + \lambda_1(1 - e^{-[R\ BSFC/(375\ \eta_p\ L/D)]}) - \lambda_2(1 - 1.024\ W_{TO}^{-0.1618}) = 0$$
$$\partial L/\partial W_{fuel} = -\lambda_1 + \lambda_2 = 0$$

Then since $\lambda_1 = \lambda_2$,

$$\lambda_1 = (e^{-[R\ BSFC/(375\ \eta_p\ L/D)]}) - 1.024\ W_{TO}^{-0.1618})^{-1} \qquad \text{(Eq.D-1)}$$

We also have

$\partial f/\partial p_1 = 0$ (W_{TO} does not depend explicitly on L/D), and

$$\partial g_1/\partial p_1 = (W_{TO}\ R\ BSFC/(375\ \eta_p\ L/D^2))\ e^{-[R\ BSFC/(375\ \eta_p\ L/D)]} \qquad \text{(Eq. D-2)}$$

Then

$$df/dp_1 = dW_{TO}/d\ L/D = -\lambda_1\ \partial g_1/\partial p_1$$

For a numerical example, let R = 25,000 mi., BSFC = 0.4, $\eta_p = 0.85$, and L/D = 20.

Then

$$W_{TO} = e^{[(25000 * 0.4/(60.675 * 0.85 * 20)) + 1.2372]}) = 55941$$

$$\lambda_1 = (e^{-[25000*0.4/(375*0.85*20)]}) - 1.024*(55941)^{-0.1618})^{-1} = 29.672$$

$$\partial g_1/\partial p_1 = (55941*25000*0.4/(375*0.85*(20)^2))*e^{-[25000*0.4/(375*0.85*20)]} = 914.06$$

and finally,

$$dW_{TO}/d\,L/D = -\lambda_1\,\partial g_1/\partial p_1 = -29.672*914.06 = -27122.0$$

The optimal sensitivity derivatives quantify the effect of changes in a design variable on the feasibility and optimality of another design decision in which that design variable appears as a parameter. For example, say a constraint that W_{TO} must be below 46,000 pounds is imposed on the choose W_{TO} and W_{fuel} design decision. Since the take-off gross weight corresponding to a lift-to-drag ratio of 20 is 55,941 pounds, the choose W_{TO} and W_{fuel} design decision is now infeasible. The optimal sensitivity derivative computed in the numerical example above can be used to approximate this situation in the choose lift-to-drag ratio design decision as follows: the desired value of W_{TO} is 46,000 pounds or less. We can approximate

$W_{TO}^*(L/D)$

= optimal (feasible) value of W_{TO} as a function of L/D

$= W_{TO}^*(20) + (dW_{TO}/dL/D)(L/D - 20)$

$= 55941 - 27122*(L/D - 20)$

This approximation is reasonably accurate, as seen in Figure D-7 (where $\Delta W_{TO} = W_{TO}^*(L/D) - 46000$), but slightly underpredicts the value of L/D required to satisfy the 46,000 pounds constraint. The linear approximation predicts that an L/D of 20.37 is required. The exact solution is closer to 20.41, but this is not a significant difference.

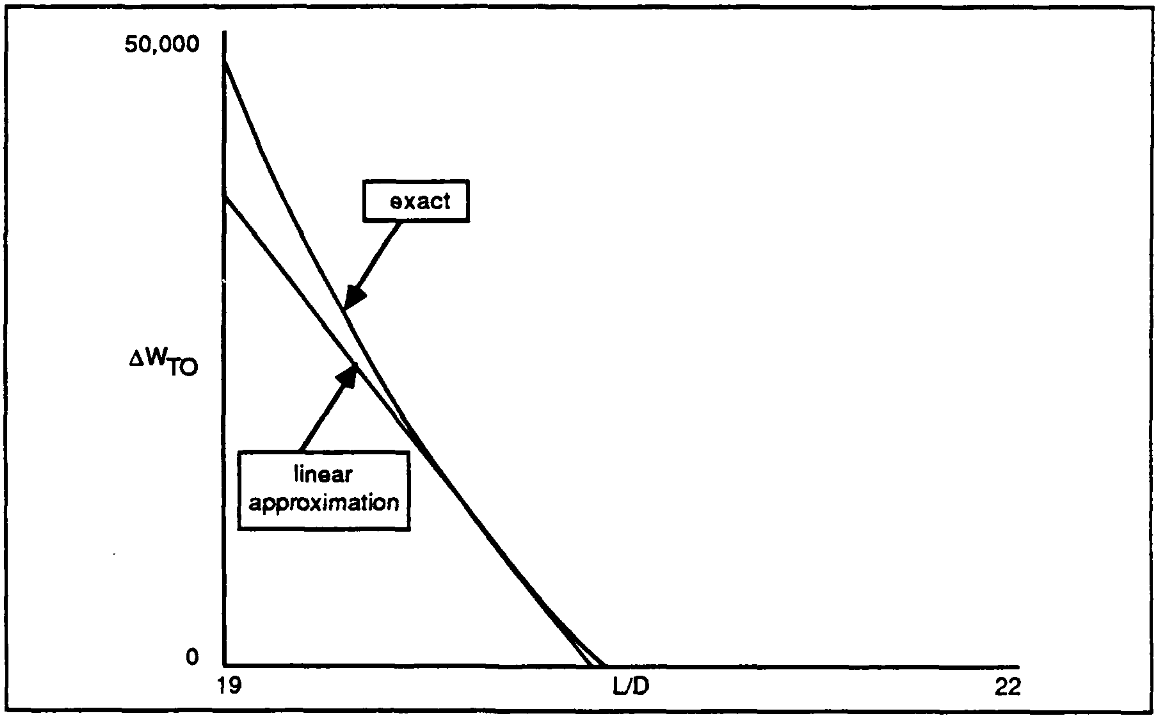


Figure D-7. Accuracy of Linear Approximation to Optimal Value of W_{TO}

4. Solution Using the Theory of Appendix B

Using the theory developed in Appendix B, we can approach the problem of balancing the goal of minimizing the L/D risk against the goal of minimizing the take-off weight. The basic idea is to place each goal in a separate optimization subproblem and then to use optimal sensitivity derivatives to balance the subproblems.

Following this method of attack, subproblems are identified,

P_1:

minimize: $f_1 = 1/(1 - (L/D/25))$

subject to:

$h_1 = L/D - p_1 = 0$

$10 \leq L/D \leq 25$

design variables: L/D

where p_1 is a parameter. Problem $\mathbf{P_2}$, associated with BSFC, will be defined after Problem $\mathbf{P_3}$:

$\mathbf{P_3}$:

$$\text{minimize: } f_3 = W_{TO}/10000$$

subject to:

$$g_1 = W_{fuel} - W_{TO}\,(1 - e^{-[R\ BSFC/(375\ h_p\ L/D)]}) \geq 0$$

$$g_2 = W_{TO} - W_{fuel} - 1.222\ W_{TO}^{0.8382} \geq 0$$

$$h_3 = L/D - p_1 = 0$$

$$h_4 = BSFC - p_2 = 0$$

design variables: $W_{TO}, W_{fuel}, L/D, BSFC$.

{side constraints which will not be used in this example}

For the moment, take $BSFC = p_2$, so that the fourth constraint in $\mathbf{P_3}$ is identically satisfied. Balancing these subproblems should correspond to the idea of balance inherent in the concept of Pareto-optimality, namely that

$$F = \omega_1 f_1(L/D) + \omega_3\, f_3(W_{TO})$$

should be minimized, with $\omega_1 + \omega_2 + \omega_3 = 1$. (Since BSFC = constant, $\omega_2 = 0$.)

From the point of view of parameter passing, the only tool we have available for balancing the subproblems is the coupling parameter p. Thus it seems natural to try to choose a value for p in such a way that F is minimized, while optimality and feasibility are maintained. With this in mind, optimal linear approximations to f_1 and f_3 are constructed, given an initial value p_1^0 for p_1 and an approximation to F in terms of the optimal linear approximations is written:

$$\begin{aligned} F_{\sim} &= \omega_1(f_1^*(p_1^0) + (df_1/dp_1)Dp_1) + \omega_3\,(f_3^*(p_1^0) + (df_3/dp_1)Dp_1) \\ &= [\omega_1(df_1/dp_1) + \omega_3\,(df_3/dp_1)]\ p_1 + \text{constant}, \end{aligned}$$

so

$$\partial F/\partial p_1 = \omega_1(df_1/dp_1) + \omega_3\,(df_3/dp_1) = 0$$

the first equality holding if all the derivatives in it are defined.

Thus, a Pareto-optimal solution corresponding to fixed values of the weightings ω when the parameter p_1 satisfies

$$\omega_1(df_1/dp_1) = -\omega_2\,(df_2/dp_1).$$

To complete this example, we verify that solutions obtained by this parameter-passing/optimal sensitivity derivative technique are in fact the Pareto-optimal ones, which can in this case be obtained explicitly (as was done earlier in this section).

An additional subproblem is associated with choosing the BSFC.

P_2:

minimize: $f_2 = 1/(BSFC/0.35) - 1)$

subject to:

$h_2 = BSFC - p_2 = 0$

$0.3501 \leq BSFC \leq 0.95$

design variable: BSFC.

For a numerical example to compare with the explicit Pareto-optimal solutions obtained earlier in this appendix, the following optimal sensitivity derivatives are needed:

$$df_1/dp_1 = \partial f_1/\partial p_1 - \mu_1\partial h_1/\partial p_1 = \mu_1$$

μ_1 can be found using the necessary condition

$$\partial f_1/\partial L/D - \mu_1\partial h_1/\partial L/D = 0$$

so $\quad \partial f_1/\partial L/D = \mu_1 = df_1/dp_1.$

Similarly, $\partial f_2/\partial BSFC = df_2/dp_2$. A straightforward argument using derivatives computed for the explicit solution yields

$$df_3/dp_1 = -\lambda_1(\partial g_1/\partial p_1)/10000$$

and

$$df_3/dp_2 = -\lambda_1(\partial g_1/\partial p_2)/10000$$

where λ_1 and $\partial g_1/\partial p_1$ given by Equations D-1 and D-2 and

$$\partial g_1/\partial p_2 = W_{TO}\,(-R/(375\,h_p\,L/D))\,e^{-[R\,BSFC/(375\,\eta_p\,L/D)]}$$

Application of Equation D-3 then gives two linear equations that can be solved for ω_1 and ω_2:

$$\omega_1 = (df_3/dp_1)(df_2/dp_2)/[(df_3/dp_2)(df_3/dp_1) - ((df_1/dp_1) - (df_3/dp_1))\,((df_2/dp_2) - (df_3/dp_2))]$$

$$\omega_2 = (\omega_1(df_3/dp_2) - (df_3/dp_2))/((df_2/dp_2) - (df_3/dp_2)).$$

These solutions can be plotted parametrically for comparison with the explicit Pareto-optimal solutions. This has been done in Figure D-8.

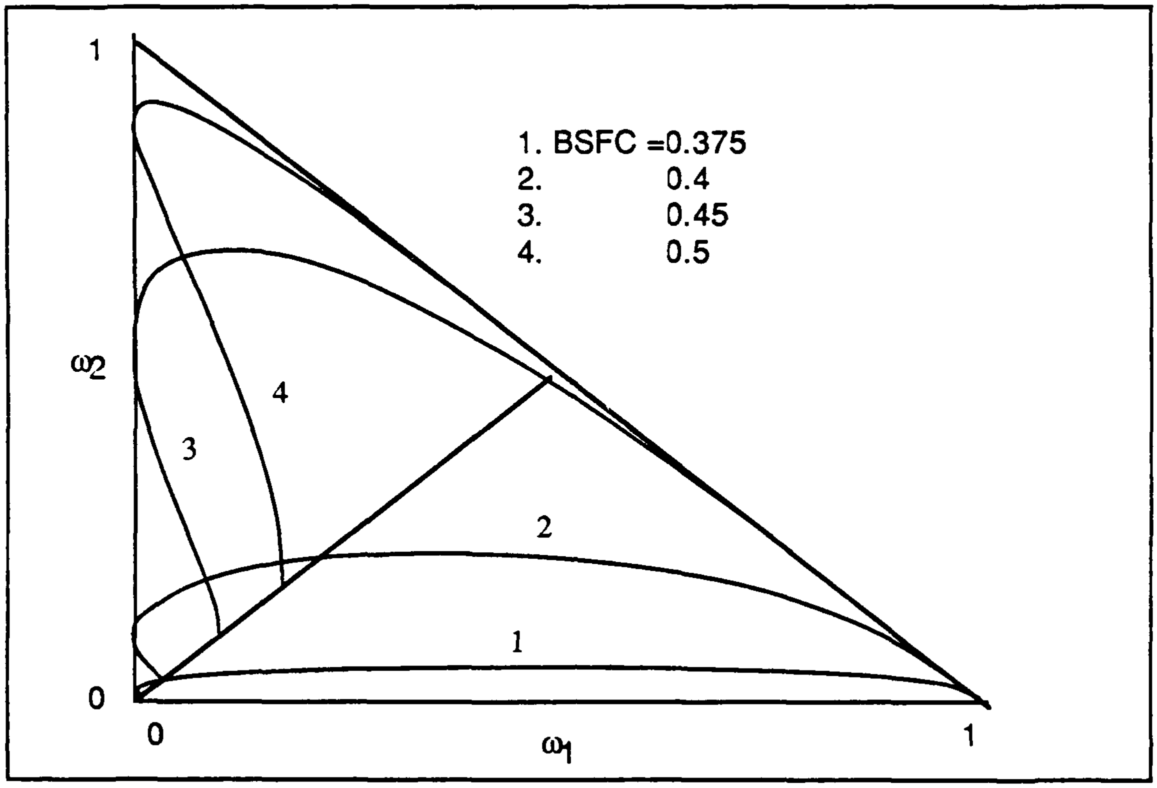


Figure D-8. Pareto-Optimal Solutions Obtained Using Optimal Sensitivity Derivatives

The results in Figure D-8 should be compared with the explicit solutions shown in Figure D-4. Clearly, the optimal sensitivity approach produces the same solutions as the explicit technique.

REFERENCES

D-1. Cohon, J.L., *Multiobjective Programming and Planning*, Academic Press, New York, 1978.

D-2. Perkins, Courtland, and Hage, *Airplane Performance, Stability and Control*.